VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Server pro podporu výyuky teorie her

Supporting Server for Games Theory Teaching

2014

Bc. Martin Huráb

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. Martin Huráb**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: Server pro podporu výuky teorie her
Supporting Server for Games Theory Teaching

Zásady pro vypracování:

V rámci série diplomových prací vzniká server pro podporu výuky teorie her. Student si na něm bude moci zahrát vybrané hry proti počítači uplatňujícímu optimální strategii, ale také si bude moci zobrazit různé výpočty a řešení úloh z oblasti teorie her. Cílem této diplomové práce je přidání základních úloh týkajících se her dvou hráčů s nulovým součtem.

Konkrétní zásady:
1. Seznamte se stručně s typy her, které se v teorii her zkoumají.
2. Podrobně se zaměřte na hry dvou hráčů s nulovým součtem.
3. S kolegy pracujícími také na vytvoření tohoto serveru si dohodněte softwarovou architekturu a vzhled, aby server působil jednotně.
4. Vytvořte část zabývající se hrami s nulovým součtem.
5. Pro matice výplatních funkcí her s nulovým součtem bude program umět hledat optimální ryzí i smíšené strategie obou hráčů (simplexovou metodou, graficky pro hry 2xM a Nx2, eliminací dominovaných strategií), případné sedlové body a cenu hry.
6. Umožněte, aby si uživatel mohl několik zvolených her zahrát proti počítači, který bude používat optimální strategii.

Seznam doporučené odborné literatury:

Thomas S. Ferguson: Game Theory, elektronický výukový text dostupný na adrese http://www.math.ucla.edu/~tom/math167.html (především Part 2)

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Ing. Martin Kot, Ph.D.**

Datum zadání: 01.09.2013
Datum odevzdání: 07.05.2014

VYSOKÁ ŠKOLA BÁŇSKÁ-TECHNICKÁ UNIVERZITA
Fakulta elektrotechniky a informatiky
1
OSTRAVA

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě dne 29. 4. 2014 Martin Huráb

Poděkování

Děkuji vedoucímu diplomové práce panu Ing. Martinu Kotovi, Ph.D. za odbornou pomoc a podnětné rady, které mi pomohly při vypracování této práce.

Abstrakt

Cílem této diplomové práce je vytvoření serveru pro podporu výuky předmětu „Teorie her a modely rozhodování v podmínkách neurčitosti“, vyučovaném na VŠB-TUO. Jelikož je výuka tohoto předmětu rozdělena do 3 částí, zaměřuje se tato práce pouze na část her s nulovým součtem, avšak výsledná webová aplikace obsahuje všechny tyto části. Tento server obsahuje kromě potřebné teorie také podrobné postupy používané při řešení probíraných typů her. Dále také uživatelům umožňuje zahrát si pár vybraných her proti počítačovému protihráči, s možností nastavení obtížnosti.

Klíčová slova

Teorie her, nulový součet, ASP.NET, webová aplikace

Abstract

The main aim of this diploma thesis is to create a web server to support the teaching of the subject „Teorie her a modely rozhodování v podmínkách neurčitosti“(„Game Theory and Decision Models in the Conditions of Indefiniteness”), which is taught at VŠB-TUO. As the tuition of this subject is divided into 3 parts, this thesis aims only at part of zero-sum games but the final web application contains all these parts. Besides the necessary theory this web server contains detailed techniques used in solving of the game types taught. It also enables the users to play some selected games against a computer opponent at user-defined difficulty.

Keywords

Game theory, zero-sum, ASP.NET, web application

Seznam použitých symbolů a zkratek

.**NET Framework 4** – základní komponenta souboru technologií společnosti Microsoft, a.s. zajišťující běh aplikací na operačních systémech Windows ve verzi 4

ASP.NET – součást .NET Framework pro tvorbu webových aplikací a služeb

CLR - Common Language Runtime

css - Cascading Style Sheets

C# - vysokoúrovňový objektově orientovaný programovací jazyk vyvinutý firmou Microsoft

gif – Graphics Interchange Format

HTML - HyperText Markup Language

IS – informační systém

XML - Extensible Markup Language

Obsah

1 Úvod

Teorie her je obor aplikované matematiky, který slouží k analýze konfliktních situací mezi jednotlivci či skupinami, kdy volba strategie jedné strany ovlivňuje výslednou výplatu druhé strany a naopak. K těmto situacím dochází nejen ve společenských hrách typu šachy či poker, jak napovídá název tohoto oboru, ale lze je také zpozorovat v ekonomii, politice, sociologii a dalších oblastech. Teorie her tyto konflikty nejen analyzuje, ale také poskytuje návod, jak mají jednotliví účastníci konfliktu své strategie volit tak, aby dosáhli co nejvyššího zisku.

Cílem této diplomové práce je navrhnout a implementovat server, který bude sloužit studentům pro snadnější pochopení látky probírané v předmětu „Teorie her a modely rozhodování v podmínkách neurčitosti“. Toho bude docíleno pomocí jednotlivých komponent aplikace, kdy si studenti budou moci nejen nastudovat potřebnou teorii, ale také zadat parametry vlastní hry, zobrazit si její řešení a následně si ji zahrát proti počítačovému protihráči.

Samotná diplomová práce je rozdělena do dvou částí. V první části (viz kapitola 2) jsou popsány základní teoretické pojmy a postupy, které teorie her používá. Prostor je zde věnován způsobům zápisu her, jejich dělení, jsou vysvětleny možnosti strategií a podrobně popsány jednotlivé metody řešení her s nulovým součtem. Ve druhé části (viz kapitola 3 až 5) je popsán návrh a implementace výsledného programového řešení. Prvně jsou zodpovězeny základní otázky k funkčnosti webové aplikace a poté je popsáno uživatelské rozhraní.

Stěžejní část představuje 5. kapitola, která popisuje celou aplikaci včetně implementace celého projektu pomocí technologie ASP.NET. Jsou zde definovány její základní funkčnosti a nástroje, včetně rozšíření Ajax Control Toolkit. Podkapitola 5.3 se pak věnuje přímo samotné aplikaci. Vysvětluje všechny početní úkony aplikace, algebraické, grafické i simplexové řešení. Dále stanovuje možnosti zadání vstupních parametrů hry či přípravy řešení hry pomocí matic nebo Kuhnova stromu. Také jsou vysvětleny principy hry proti počítači či simulované hry. V neposlední řadě je součástí kapitoly také popis jednotlivých her, konkrétně Souboj prvočísel, Dobývání hradu a Kerbindak.

Záměrem této práce je naprogramovat internetovou aplikaci pro řešení her s nulovým součtem tak, aby mohla být doporučena a užívána při výuce předmětu „Teorie her a modely rozhodování v podmínkách neurčitosti“ i při samostudiu. Součástí aplikace je také uživatelská příručka, která umožňuje plně si osvojit ovládání všech komponent. V minulosti již vznikla aplikace pro část zabývající se kombinatorickými hrami [5] a souběžně s touto prací vzniká aplikace pro řešení her s nenulovým součtem [1]. Všechny tyto 3 části budou součástí jedné webové aplikace. Úspěšnost tohoto softwarového řešení bude prokázána jejím využíváním studenty.

2 Teorie her

V této kapitole jsou obsaženy základní pojmy používané v teorii her.

2.1 Způsoby zápisu her

Tato kapitola popisuje základní rozdělení zápisu her.

2.1.1 Hra v normálním tvaru

Hra v normálním tvaru (nebo též strategickém tvaru) je prvním ze základních matematických modelů teorie her. Je určena třemi množinami (N, A, f), kde

- $N = \{1, 2, \dots, n\}$ je množina hráčů,
- $A = \{A_1, \dots, A_n\}$ je množina prostorů strategií hráčů,
- $f = \{f_1(a_1, \dots, a_n), \dots, f_n(a_1, \dots, a_n)\}$ je množina výplatních funkcí hráčů.

Každý z hráčů 1 … n vybírá jednu ze svých strategií $a_i \in A_i$. Hráči volí své strategie současně, což zaručuje, že mají všichni hráči stejné informace a žádný z nich není zvýhodněn tím, že by mohl reagovat na výběr protihráče. Průnikem těchto strategií získáme hodnotu výplatní funkce $f_n(a_i)$ pro jednotlivé hráče.

Množina N obsahuje nejčastěji dva hráče, jelikož při tomto počtu lze výplatní funkce hráčů vyjádřit tabulkou, kde řádky odpovídají strategiím hráče I a sloupce strategiím hráče II.

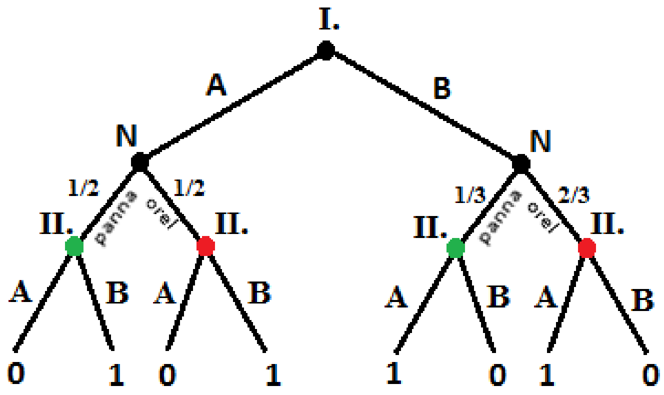
Příklad 1 je zadán maticí výplatních funkcí pro obecně známou hru „Kámen, nůžky, papír“, kde každý z hráčů vybírá jednu ze strategií „Kámen“, „Nůžky“, nebo „Papír“. Jedná se o hru s nulovým součtem dvou hráčů, kdy je výhra symbolizována ziskem jednoho bodu, remíza ziskem 0 bodů a prohra ztrátou jednoho bodu. Matice je zapsána z pohledu hráče I.

		Hráč II		
	Strategie	Kámen	Nůžky	Papír
Hráč I	Kámen	0	1	-1
	Nůžky	-1	0	1
	Papír	1	-1	0

Příklad 1

2.1.2 Hra v rozvinutém tvaru

Hra v rozvinutém tvaru (nebo též explicitním tvaru) na rozdíl od hry v normálním tvaru předpokládá, že hráči rozhodují konfliktní situaci po tazích a jejich rozhodnutí závisí na předchozích rozhodnutích ostatních hráčů. Tato rozhodnutí jednotlivých hráčů se dají zachytit pomocí tzv. Kuhnova stromu. Kuhnův strom umožňuje kromě zápisu posloupnosti tahů hráčů také zachytit hráče, označovaného jako Příroda, což je náhodný prvek, jenž se v mnoha hrách vyskytuje. Ukázka Kuhnova stromu je zobrazena níže na Obr. 1. Zachycuje jednoduchou hru 2 hráčů (příklad 2), kde hráč I vybere 1 ze 2 mincí. Mince A je poctivá, což znamená, že padnutí panny i orla mají stejnou pravděpodobnost, tedy $^1/_2$ každá. Naopak mince B je nepoctivá s pravděpodobností padnutí panny $^1/_3$ a padnutí orla $^2/_3$. Hráč I od sebe tyto 2 mince rozezná. Po výběru mince hráč I touto mincí hodí. Nyní je na tahu hráč II, jehož úkolem je uhodnout, kterou z mincí hráč I hodil. Pokud hráč II minci označí správně, je výplata 0, v opačném případě hráč II zaplatí hráči I 1$. V tomto stromě jsou barevně rozlišeny informační množiny, které stanovují, do jaké míry je hráč II schopen určit, ve kterých uzlech se při rozhodování ve stromě nachází.



Obr. 1 - Příklad 2 - Kuhnův strom

2.2 Dělení her podle výher

Tato kapitola popisuje základní rozdělení her podle výher.

2.2.1 Hra s nulovým součtem

Hra s nulovým součtem je hra, ve které je suma výplatních funkcí všech hráčů pro vybranou strategii rovna nule, tedy

$$\sum_{i=1}^{n} f_i(a_1, a_2, \dots, a_n) = 0,$$

pro $a_1 \in A_1$, $a_2 \in A_2$, ..., $a_n \in A_n$.

To znamená, že suma výher hráčů majících v daném kole zisk, je rovna výši sumy ztrát hráčů, kteří jsou v daném kole ve ztrátě.

2.2.2 Hra s nenulovým součtem

Hra s nenulovým součtem, na rozdíl od hry s nulovým součtem, umožňuje různě vysoké sumy výplatních funkcí. Z tohoto důvodu mohou být některé strategie výhodné pro všechny hráče, a tudíž má smysl uvažovat o jejich spolupráci. O hrách s nenulovým součtem tato práce nepojednává, avšak zabývá se jimi práce kolegy Bc. Marka Záškodného [1], která vznikala souběžně s touto prací.

2.3 Hra 2 hráčů s nulovým součtem

Tato kapitola popisuje pojmy a metody, týkající se her 2 hráčů s nulovým součtem.

2.3.1 Sedlový bod

Sedlový bod a_{ij} je takový prvek matice, který splňuje následující dvě podmínky:

- prvek a_{ij} je roven minimální hodnotě řádku *i*,
- prvek a_{ij} je roven maximální hodnotě sloupce *j*.

Hraní strategie, obsahující sedlový bod, hráči zaručí, že získá nejméně takovou částku, jakou sedlový bod obsahuje. Pokud se jeden z hráčů bude držet své strategie, obsahující sedlový bod, a druhý nikoliv, může si hráč dodržující svou optimální strategii pouze polepšit. Na Obr. 2 je k vidění výplatní matice *A* hry 2 hráčů s nulovým součtem, která sedlový bod obsahuje. Konkrétně se nachází ve druhém řádku čtvrtého sloupce a má hodnotu -3.

$$\begin{array}{rcc} & & \text{ř. min} \\ A = & \begin{bmatrix} 5 & 5 & -7 & -4 \\ -2 & 4 & 7 & -3 \\ 5 & -4 & 1 & -3 \\ 7 & -6 & 12 & -5 \end{bmatrix} & \begin{matrix} -7 \\ -3 \\ -4 \\ -6 \end{matrix} \\ \text{sl. max} & \begin{matrix} 7 & 5 & 12 & -3 \end{matrix} & \end{array}$$

Obr. 2 - Sedlový bod

2.3.2 Cena hry

Cena hry je průměrný zisk (ztráta), který hráč získá po dostatečném počtu odehraných her. Obvykle je udávána z pohledu hráče I, kdy kladná hodnota znamená zisk a záporná ztrátu. V případě, že hra obsahuje sedlový bod, je cena hry rovna hodnotě tohoto bodu. V ostatních případech se její průměrná hodnota určuje pomocí metod popsaných v kapitole 2.4.

2.3.3 Typy hráčů

Inteligentní (racionální) hráč je takový hráč, který vždy volí své tahy tak, aby dosáhl co nejvyššího zisku (případně co nejnižší ztráty). Naopak **neinteligentní hráč** volí své tahy náhodně. Jelikož teorie her předpokládá, že jsou všichni hráči účastnící se hry racionální, vystupuje neinteligentní hráč jako zástupce náhodného mechanizmu.

2.3.4 Ryzí strategie

Pojem ryzí strategie označuje typ strategie, kdy hráči ze svých strategií vybírají pouze jednu nejvýhodnější, kterou poté uplatňují v každém tahu. Pokud hrají oba hráči každé kolo pouze své ryzí strategie, je cena hry konstantní.

Na Obr. 3 vidíme výplatní matici příkladu 3. Ryzí strategií hráče I (vybírajícího řádky) je v tomto příkladu třetí řádek, jehož minimální hodnota (2) je nejvyšší ze všech minimálních řádkových hodnot. Ryzí strategie hráče II (vybírajícího sloupce) je první sloupec s maximální hodnotou 3, která je ze všech maximálních sloupcových hodnot nejnižší.

$$A = \begin{bmatrix} 1 & 4 & 5 \\ 3 & -2 & 6 \\ 3 & 2 & 6 \end{bmatrix}$$

Obr. 3 - Příklad 3 - ryzí strategie

2.3.5 Smíšená strategie

Smíšená strategie označuje vektor o velikosti počtu strategií hráče, jehož prvky představují pravděpodobnosti volby jednotlivých ryzích strategií. Tento vektor pravděpodobností hráči zaručí, že v průměru dosáhne nejvyššího možného zisku. Tyto prvky mohou nabývat hodnot z intervalu $\langle 0; 1 \rangle$ a jejich součet musí být roven 1. Pokud hra obsahuje sedlový bod, získá strategie obsahující sedlový bod pravděpodobnost 1, všechny ostatní strategie získají pravděpodobnost 0. O způsobech určení vektoru smíšených strategií pojednává kapitola 2.4.

$$\sum_{i=1}^{n} a_1, a_2, \dots, a_n = 1,$$

Smíšenou strategií pro příklad 3 je pro hráče I vektor $(1/_4, 0, 3/_4)$, pro hráče II je to poté vektor $(1/_2, 1/_2, 0)$.

2.3.6 Dominování

Dominování je název metody sloužící k odstranění strategií, jejichž hraní by nebylo pro hráče výhodné, ať už by protihráč zvolil jakoukoli ze svých strategií. Účelem dominování je redukovat rozměry matice strategií a zjednodušit tak výpočet pro nalezení optimálních strategií, přičemž je matice vzniklá dominováním ekvivalentní s původní maticí. Dominování můžeme rozdělit na silné (striktní) a slabé dominování.

2.3.6.1 Silné (striktní) dominování

Silné dominování zaručuje, že je možné dominovanou strategii z množiny strategií vypustit, jelikož její hraní by hráči pro každou hodnotu výplatní funkce nepřineslo vyšší zisk, než hraní dominující strategie. Při zkoumání, zda matice obsahuje dominované strategie, zjišťujeme, jestli některý z řádků matice (r_a) obsahuje vyšší hodnoty ve všech příslušných sloupcích, než řádek r_b, nebo také, zda matice obsahuje sloupec c_a, který obsahuje nižší hodnoty než sloupec c_b ve všech příslušných řádcích. Pokud takovýto řádek (sloupec) nalezneme, můžeme jej vypustit, aniž bychom přišli o některé možné řešení.

Striktní dominování pro příklad 3:

$$A = \begin{bmatrix} 1 & 4 & 5 \\ 3 & -2 & 6 \\ 3 & 2 & 6 \end{bmatrix} \longrightarrow A' = \begin{bmatrix} 1 & 4 \\ 3 & -2 \\ 3 & 2 \end{bmatrix}$$

Obr. 4 - Striktní dominování

Vidíme, že všechny hodnoty ve třetím sloupci matice *A* jsou vyšší než hodnoty v prvním sloupci, proto říkáme, že je třetí sloupec striktně dominovaný prvním sloupcem a můžeme jej z matice vypustit. Nově vzniklou matici o třech řádcích a dvou sloupcích označíme *A'*.

2.3.6.2 Slabé dominování

Slabé dominování odstraní méně přínosné strategie, avšak může způsobit odstranění některých možných řešení. Postupujeme podobně jako při odstraňování silně dominovaných řádků a sloupců, nyní však nemusí být hodnoty vybraného řádku r_a ostře větší, než hodnoty řádku r_b, případně hodnoty vybraného sloupce c_a nemusí být ostře menší než hodnoty sloupce c_b. Postačí, když budou tyto hodnoty větší nebo rovny pro řádky a nižší nebo rovny pro sloupce.

Slabé dominování pro matici *A'* pro příklad 3:

$$A' = \begin{bmatrix} 1 & 4 \\ \cancel{3} & \cancel{-2} \\ 3 & 2 \end{bmatrix} \longrightarrow A'' = \begin{bmatrix} 1 & 4 \\ 3 & 2 \end{bmatrix}$$

Obr. 5 - Slabé dominování

Hodnoty ve druhém řádku jsou nižší nebo rovny hodnotám ve třetím řádku. Při použití slabého dominování můžeme druhý řádek z matice vypustit. Matice má po odstranění dominovaných strategií pouze dva řádky a dva sloupce, což značně usnadní nalezení smíšených strategií. Tuto novou matici typu 2×2 označíme *A*'‘.

2.3.6.3 Složené dominování

V některých případech není strategie dominována (ať už slabě či silně) jednou strategií, ale kombinací několika strategií. Pokud budeme tyto strategie hrát s určitou pravděpodobností, získáme ve všech případech lepší zisky, než pokud bychom hráli takto dominovanou strategii.

Složené dominování si opět ukážeme na příkladu, jehož výplatní matice je na Obr. 6. Mějme hru 2 hráčů s nulovým součtem zadanou následující výplatní maticí *A*.

$$A = \begin{bmatrix} -3 & 7 & 2 \\ -2 & 2 & 6 \\ 4 & -6 & 3 \end{bmatrix}$$

Obr. 6 - Zadání příkladu 4

Matice neobsahuje sedlový bod ani žádnou strategii, která by byla silně či slabě dominovaná jinou strategií. Pokud však vynásobíme prvky prvního a druhého sloupce koeficientem $^1/_2$, zjistíme, že součty hodnot těchto dvojic prvků jsou vždy nižší nebo rovny hodnotám prvků ve třetím sloupci, čili je třetí sloupec slabě dominovaný kombinací prvního a druhého sloupce. Dále vidíme, že je v nově vzniklé matici druhý řádek silně dominovaný kombinací prvního a třetího řádku, konkrétně koeficientem $^2/_3$ pro první řádek a $^1/_3$ pro třetí řádek. Nově vzniklou matici o rozměrech 2×2 označíme *A*‘.

$$A = \begin{array}{c} \begin{array}{ccc} 1/2 & 1/2 & \end{array} \\ \begin{bmatrix} -3 & 7 & \cancel{2} \\ -2 & 2 & \cancel{6} \\ 4 & -6 & \cancel{3} \end{bmatrix} \end{array} \longrightarrow \begin{bmatrix} -3 & 7 \\ \cancel{-2} & \cancel{2} \\ 4 & -6 \end{bmatrix} \begin{array}{l} 2/3 \\ \\ 1/3 \end{array} \longrightarrow A' = \begin{bmatrix} -3 & 7 \\ 4 & -6 \end{bmatrix}$$

Obr. 7 - Složené dominování

2.4 Metody řešení her s nulovým součtem 2 hráčů

Hry dvou hráčů s nulovým součtem, které jsou zadány ve strategickém tvaru, je možné podle počtu strategií hráčů rozdělit do 3 kategorií:

- Hry 2×2,
- Hry 2×n, m×2,
- Hry m×n.

První z dvojice čísel udává počet strategií prvního hráče, druhé počet strategií druhého hráče. Před samotným řešením je nutné otestovat, zda matice neobsahuje sedlový bod. V případě nalezení sedlového bodu výpočet končí a řešením jsou ryzí strategie obsahující tento bod. Dále je vhodné pokusit se najít a odstranit dominované strategie, což může rozměry matice snížit a výpočet tím zjednodušit, případně snížit rozměry matice na 2×2, u níž je výpočet nejsnazší.

2.4.1 Hry typu 2×2

Hra, obsahující matici o dvou řádcích a dvou sloupcích je nejjednodušším případem (pomineme-li případ, kdy by měl jeden z hráčů k dispozici pouze jednu strategii), jaký může při řešení nastat. Pro názornost si jednotlivé prvky matice *A* označíme písmeny *a*, *b*, *c*, *d*.

$$A = \begin{bmatrix} a & b \\ d & c \end{bmatrix}$$

Obr. 8 - Obecná matice algebraického řešení

Jelikož má hráč I na výběr pouze ze dvou strategií, volí svou první strategii s pravděpodobností *p* a druhou s pravděpodobností 1 - *p*. Obdobně hráč II volí svou první strategii s pravděpodobností *q* a druhou s pravděpodobností 1 - *q*.

Pokud předpokládáme, že žádný z prvků není sedlovým bodem, pak musí platit:

Pokud je $a \geq b$, pak $b < c$,

protože je $b < c$, tak $c > d$,

protože $a \geq b$ a současně $c > d$, tak $d < a$,

protože $d < a$, tak $a > b$,

tedy pokud $a \geq b$, tak $a > b < c > d$.

Pokud je $a \leq b$, pak $b > c$,

protože je $b > c$, tak $c < d$,

protože $a \leq b$ a současně $c < d$, tak $d > a$,

protože $d > a$, tak $a < b$,

tedy pokud $a \leq b$, tak $a < b > c < d$.

Díky těmto znalostem získáme rovnici pro výpočet smíšených strategií obou hráčů. Nejprve vyjádříme strategii pro hráče I:

$$ap + d(1 - p) = bp + c(1 - p), \tag{1}$$

z této rovnice dostáváme

$$p = \frac{c - d}{a - b + c - d}$$

Pro hráče II získáme rovnici

$$aq + b(1 - q) = dq + c(1 - q), \tag{2}$$

z níž vyjádříme q:

$$q = \frac{c - b}{a - b + c - d}$$

Průměrnou cenu hry v můžeme získat např. z rovnice (1):

$$v = ap + d(1 - p) = \frac{ac - bd}{a - b + c - d} \tag{3}$$

Výpočet si ukážeme na příkladu (Obr. 9), kde je výplatní matice následovná:

$$A = \begin{bmatrix} -1 & 4 \\ 3 & -2 \end{bmatrix}$$

Obr. 9 - Výplatní matice příkladu 5

Nejprve zjistíme, zda matice obsahuje sedlový bod. Jelikož matice neobsahuje žádný bod, který by byl minimem řádku a současně maximem sloupce, můžeme ve výpočtu pokračovat. Pomocí obecné matice (1) vyjádříme prvky *a*, *b*, *c*, *d*.

a = -1; b = 4; c = -2; d = 3.

Následně můžeme z výše vyjádřených rovnic vypočítat smíšené strategie obou hráčů a cenu hry.

$$p = \frac{c-d}{a-b+c-d} = \frac{-2-3}{-1-4+(-2)-3} = 0{,}5$$
$$q = \frac{c-b}{a-b+c-d} = \frac{-2-4}{-1-4+(-2)-3} = 0{,}6$$
$$v = \frac{ac-bd}{a-b+c-d} = \frac{-1.(-2)-4.3}{-1-4+(-2)-3} = 1$$

$p = 0{,}5 \Rightarrow 1 - p = 0{,}5$

$q = 0{,}6 \Rightarrow 1 - q = 0{,}4$

Optimální smíšená strategie hráče I je vektor (0,5; 0,5).

Optimální smíšená strategie hráče II je vektor (0,6; 0,4).

Cena hry je 1.

2.4.2 Hry typu 2×n a m×2

Hry, ve kterých má jeden z hráčů na výběr pouze ze dvou strategií, lze vyřešit pomocí grafické metody. Základem této metody jsou lineární funkce, které získáme z výplatní matice. Budeme-li uvažovat hru typu 2×n, kde hráč I (vybírající řádky) volí první strategii s pravděpodobností p a druhou s pravděpodobností 1 - p, jsou funkce v následujícím tvaru:

$$a_{11}(p) + a_{21}(1 - p),\ a_{12}(p) + a_{22}(1 - p),\ \ldots,\ a_{1n}(p) + a_{2n}(1 - p),$$

kde a_{ij} je prvek matice na pozici ij.

U her m×2, kde hráč II volí svou první strategii s pravděpodobností q a druhou strategii s pravděpodobností 1 – q, jsou to funkce ve tvaru:

$$a_{11}(q) + a_{12}(1 - q),\ a_{21}(q) + a_{22}(1 - q),\ \ldots,\ a_{m1}(p) + a_{m2}(1 - q),$$

kde a_{ij} je prvek matice na pozici ij.

Horizontální osa představuje rozložení pravděpodobnosti hraní vybrané strategie a nabývá hodnot z intervalu $\langle 0; 1\rangle$. Levá vertikální osa znázorňuje situaci, kdy hráč (u her 2×n hráč I, u her m×2 hráč II) použije svou druhou strategii, pravá vertikální osa naopak situaci, kdy použije svou první strategii.

Pro **hry 2×n** vyneseme na příslušné vertikální osy odpovídající hodnoty z výplatní matice, tedy z prvního řádku na pravou osu, z druhého řádku na levou osu. Poté pro j=1, 2, …, n úsečkami propojíme všechny dvojice bodů a_{1j} a a_{2j} a nalezneme dolní obálku grafu. Dolní obálka reprezentuje nejnižší možné zisky obdržené při hraní jednotlivých strategií bez ohledu na volby protihráče. Jelikož je hráč I racionální, snaží se nalézt nejvyšší z těchto nejnižších zisků. Nejvyšším ziskem je průsečík úseček, nacházející se v tomto grafu nejvýše. Tento průsečík označíme jako maximinový bod. Maximinový bod nám poslouží k určení optimální smíšené strategie. Spustíme-li z tohoto bodu kolmici k horizontální ose, získáme v průsečíku této kolmice a horizontální osy hodnotu p odpovídající pravděpodobnosti hraní první strategie hráče I. Hraní druhé strategie určíme snadno ze vztahu výše, tedy $1 - p$.

Pro **hry m×2** vyneseme hodnoty z prvního sloupce matice na pravou osu a hodnoty z druhého sloupce na levou osu. Pro i=1, 2, …, m propojíme úsečkami všechny dvojice bodů a_{i1} a a_{i2} a nalezneme horní obálku grafu. Tak jako u her 2×n určovala dolní obálka nejnižší možné zisky hráče I, určuje horní obálka grafu u her m×2 nejvyšší možné ztráty hráče II. Pro hráče II je samozřejmě nejvýhodnější mít co nejnižší ztrátu, proto v grafu hledá takový průsečík úseček, který je nejníže. Tomuto bodu říkáme minimaxový bod. Pomocí tohoto bodu nalezneme optimální strategii q podobně jako u her 2×n, zjistíme tedy hodnotu tohoto bodu na horizontální ose.

Pro určení optimální smíšené strategie hráče, jenž měl ve hře na výběr z více než 2 strategií (u her 2×n hráč II, u her m×2 hráč I) se opět zaměříme na nalezený maximinový, případně minimaxový bod. Úsečky procházející tímto bodem vymezují strategie, jejichž hraní zajistí tomuto hráči nejvyšší možný průměrný zisk, respektive nejnižší průměrnou ztrátu. Původní matici tak můžeme zredukovat na matici o rozměrech 2×2 tím, že použijeme pouze ty strategie, jejichž úsečky procházely tímto bodem. Matici o rozměrech 2×2 poté můžeme vyřešit například pomocí algebraického řešení (viz 2.4.1).

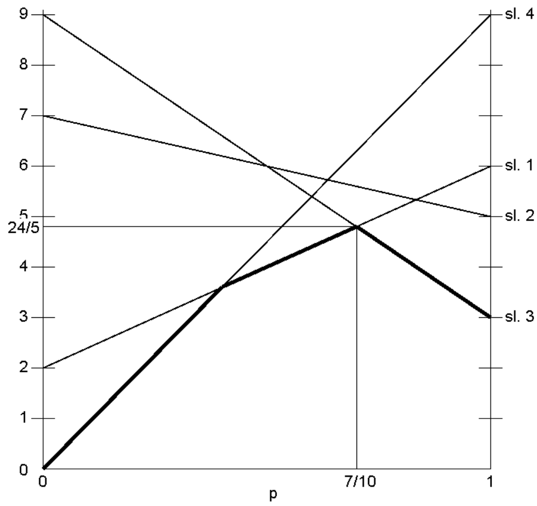
Cenu hry získáme opět z maximinového či minimaxového bodu, konkrétně ji reprezentuje jeho vertikální souřadnice.

Vše si opět ukážeme na příkladu (Obr. 10), který popisuje řešení hry 2×n, zadaného výplatní maticí A.

$$A = \begin{bmatrix} 6 & 5 & 3 & 9 \\ 2 & 7 & 9 & 0 \end{bmatrix}$$

Obr. 10 - Výplatní matice příkladu 6

Uplatněním pravidel popsaných výše získáme následující graf:



Obr. 11 - Grafické řešení

Tučně je zde zvýrazněna dolní obálka, jejímž nejvyšším bodem je maximinový bod se souřadnicemi $\left[{}^{7}/_{10}; {}^{24}/_{5}\right]$. Hodnota ${}^{7}/_{10}$ je tedy optimální pravděpodobnost hraní první strategie hráčem I, druhý řádek bude hráč I hrát s pravděpodobností $1 - {}^{7}/_{10} = {}^{3}/_{10}$. Smíšená strategie hráče I je tedy dána vektorem $({}^{7}/_{10}; {}^{3}/_{10})$.

Samozřejmě nás zajímají i smíšené strategie hráče II. Vidíme, že maximinovým bodem procházejí strategie č. 1 a 3 hráče II. Tyto 2 strategie zapíšeme do nové matice *A'*:

$$A' = \begin{bmatrix} 6 & 3 \\ 2 & 9 \end{bmatrix}$$

Obr. 12 - Výplatní matice pro nalezení strategie hráče II.

Použitím metody pro řešení her 2×2 získáme pro hráče II následující vektor smíšených strategií: $\left({}^{3}/_{5}; {}^{2}/_{5}\right)$. Jelikož obsahovala původní matice 4 sloupce, musí výsledný vektor obsahovat opět 4 hodnoty. Strategie, které jsme z původní matice vyloučili, získají pravděpodobnost 0. Výsledná smíšená strategie je tedy $\left({}^{3}/_{5}; 0; {}^{2}/_{5}; 0\right)$.

Cenu hry můžeme vyčíst z grafu. Je to vertikální souřadnice maximinového bodu, tj. ${}^{24}/_{5}$.

2.4.3 Hry typu m×n

Hry, jejichž výplatní matice obsahuje m řádků a n sloupců, lze řešit za pomoci simplexové metody. Simplexová metoda je iterativní postup mající za cíl vyřešit úlohy lineárního programování.

Postup řešení pomocí simplexové tabulky podle [2]:

1. Pokud obsahuje matice záporné hodnoty, je nutné ke všem prvkům přičíst konstantu c o velikosti nejmenšího prvku v matici. Tímto zajistíme, že budou všechny prvky nenulové. Pokud tento krok učiníme, musíme v kroku 7. od výsledné ceny hry tuto konstantu opět odečíst.

2. Vytvoříme tabulku o velikosti m×n, kterou vyplníme prvky z původní matice $a_{11}, \ldots, a_{mn}$, případně její upravené hodnoty z kroku 1. Tuto tabulku rozšíříme tak, že do záhlaví řádků přidáme proměnné hráče I označené $x_1, x_2, \ldots, x_m$ a do záhlaví sloupců proměnné hráče II označené $y_1, y_2, \ldots, y_n$. Dále přidáme řádek z, do jehož prvků zapíšeme hodnotu -1 a sloupec w, do jehož prvků zapíšeme hodnotu 1. Na pozici [m + 1, n + 1], tj. ve spodním pravém rohu tabulky, zapíšeme hodnotu 0. Takto vytvořená tabulka vypadá následovně:

	$\mathbf{y_1}$	$\mathbf{y_2}$	…	$\mathbf{y_n}$	**w**
$\mathbf{x_1}$	a_{11}	a_{12}	…	a_{1n}	1
$\mathbf{x_2}$	a_{21}	a_{22}	…	a_{2n}	1
⋮	⋮	⋮		⋮	⋮
$\mathbf{x_m}$	a_{m1}	a_{m2}	…	a_{mn}	1
z	-1	-1	…	-1	0

Tabulka 1

3. V tabulce nalezneme pivotní prvek o souřadnicích [p, q]. Hodnotu q určíme tak, aby prvek o souřadnicích [z, q] obsahoval zápornou hodnotu. Pokud se v tabulce takovýchto prvků nachází více, vybereme libovolný z nich. Pivot musí vždy obsahovat kladnou hodnotu, proto se při hledání souřadnice p zaměříme jen na řádky, jejichž hodnota je ve sloupci q kladná. Hodnotu souřadnice p určíme tak, aby byl podíl dělence a[p, w] a dělitele a[p, q] nejnižší.

4. Nové hodnoty prvků a_{ij}, kde $i \neq p$ a $j \neq q$, určíme podle vztahu $a_{ij} = a_{ij} - \frac{a_{pj} \cdot a_{iq}}{a_{pq}}$,

hodnoty prvků a_{pj}, kde $j \neq q$, určíme ze vztahu $a_{pj} = \frac{a_{pj}}{a_{pq}}$,

hodnoty prvků a_{iq}, kde $i \neq p$, získáme ze vztahu $a_{iq} = -\frac{a_{iq}}{a_{pq}}$,

novou hodnotu prvku na pozici pivota poté získáme jeho převrácenou hodnotou, tedy $a_{pq} = \frac{1}{a_{pq}}$.

5. Záhlaví řádku na pozici pivota p vyměníme se záhlavím sloupce na pozici pivota q.

6. Pokud se v řádku z nachází alespoň 1 záporná hodnota, vrátíme se ke kroku 3.

7. Nenachází-li se v řádku z již žádná záporná hodnota, můžeme určit řešení. Smíšenou strategii hráče I určíme následovně: Proměnné hráče I ($x_1, x_2, \ldots, x_m$), které se v poslední tabulce nacházejí v záhlaví řádků, obdrží pravděpodobnost 0. Ostatní, tedy ty, jež se nacházejí v záhlaví sloupců, obdrží pravděpodobnost o velikosti podílu prvku a_{zj} a a_{zw}, kde j je sloupec s proměnnou hráče I v záhlaví. Proměnné hráče II. ($y_1, y_2, \ldots, y_n$), nacházející se v poslední tabulce v záhlaví sloupců, obdrží pravděpodobnost 0. Ostatní, nacházející se v záhlaví řádků, získají pravděpodobnost o velikosti podílu prvku a_{iw} a a_{zw}, kde i je řádek obsahující proměnnou hráče II v záhlaví. Cena hry v se určí převrácenou hodnotou prvku a_{zw}. Pokud jsme v kroku 1. přičetli k prvkům konstantu c, odečteme tuto konstantu od výsledné ceny hry v, tedy $v = \frac{1}{a_{zw}} - c$.

Aplikaci této metody si pro jednoduchost výpočtu předvedeme na příkladu hry, která je zadaná maticí A o rozměrech 2×2 a neobsahuje žádné dominované strategie. Tento příklad jsme již řešili v kapitole 2.4.1 (příklad 5).

$$A = \begin{bmatrix} -1 & 4 \\ 3 & -2 \end{bmatrix}$$

Obr. 13 - Výplatní matice pro příklad 5

Prvním krokem je přičtení konstanty c ke všem prvkům matice tak, aby žádný z nich nebyl záporný. Pro tento případ je $c = 2$. Tím získáme matici A'.

$$A' = \begin{bmatrix} 1 & 6 \\ 5 & 0 \end{bmatrix}$$

Obr. 14 - Upravená výplatní matice pro příklad 5

Pokračujeme krokem 2, kdy z matice A' vytvoříme první simplexovou tabulku. V této tabulce následně určíme pivotní prvek. Jelikož jsou hodnoty z pro obě proměnné y_1, y_2 záporné, můžeme jako pivotní sloupec zvolit libovolný z nich. Vybereme si tedy první sloupec, $q = 1$. Pivotní řádek vybereme nalezením minima z podílů $1/_1$ a $1/_5$ ⇒ nejnižší hodnotu obsahuje druhý řádek, proto $p = 2$. Pivotní prvek v tabulce označíme symbolem *.

	y_1	y_2	w
x_1	1	6	1
x_2	5*	0	1
z	-1	-1	0

Tabulka 2

Nyní vytvoříme novou tabulku podle pravidel popsaných výše. Záhlaví řádku p vyměníme se záhlavím řádku q, tedy proměnnou x_2 vyměníme s proměnnou y_1. Všimněme si, že se hodnota prvku a_{zw} zvýšila z 0 na ${}^{1}/_{5}$, což značí nalezení lepšího řešení. Jelikož již zápornou hodnotu z obsahuje pouze druhý sloupec, je druhý sloupec pivotní, $q = 2$. Určení pivotního řádku je také snadné, protože kladnou hodnotu obsahuje pouze první řádek, proto $p = 1$.

	x_2	y_2	w
x_1	$-{}^{1}/_{5}$	6*	${}^{4}/_{5}$
y_1	${}^{1}/_{5}$	0	${}^{1}/_{5}$
z	${}^{1}/_{5}$	-1	${}^{1}/_{5}$

Tabulka 3

Po vytvoření nové tabulky jsou již všechny hodnoty z nezáporné, tudíž výpočet končí a zbývá jen určit optimální smíšenou strategii pro oba hráče a cenu hry.

	x_2	x_1	w
y_2	$-{}^{1}/_{30}$	${}^{1}/_{6}$	${}^{2}/_{15}$
y_1	${}^{1}/_{5}$	0	${}^{1}/_{5}$
z	${}^{1}/_{6}$	${}^{1}/_{6}$	${}^{1}/_{3}$

Tabulka 4

Tu určíme postupem z kroku 7. Žádná z proměnných hráče I (x_1, x_2) se v poslední tabulce nenachází v záhlaví řádku, proto je optimální smíšená strategie hráče I dána vektorem $\left(\frac{{}^{1}/_{6}}{{}^{1}/_{3}}, \frac{{}^{1}/_{6}}{{}^{1}/_{3}}\right) = \left(\frac{1}{2}, \frac{1}{2}\right)$. Podobná situace je i s proměnnými hráče II (y_1, y_2), žádná se nenachází v záhlaví sloupce, proto je optimální smíšená strategie $\left(\frac{{}^{1}/_{5}}{{}^{1}/_{3}}, \frac{{}^{2}/_{15}}{{}^{1}/_{3}}\right) = \left(\frac{3}{5}, \frac{2}{5}\right)$. Cena hry v je $\frac{1}{{}^{1}/_{3}} - 2 = 1$. Optimální strategie i cena hry nám vyšly stejně jako při algebraickém řešení, z čehož lze usuzovat, že je výpočet správný.

3 Analýza požadavků

Tato kapitola se věnuje analýze požadavků nově vzniklého softwarového řešení. Jelikož je výstupem této diplomové práce serverová aplikace kladoucí si za cíl pomoci studentům předmětu Teorie her při jejich studijním úsilí, vycházel jsem v této analýze, kromě požadavků vedoucího práce, také ze svých vlastních zkušeností s tímto předmětem.

3.1 Funkční požadavky

Základní funkční požadavky byly vymezeny v zadání práce, další, specifičtější, byly postupně přidávány při konzultacích s vedoucím práce.

3.1.1 PROČ nový systém

Jak již bylo zmíněno v úvodu této kapitoly, vzniká tento systém z důvodu usnadnění pochopení látky předmětu Teorie her, konkrétně části věnované hrám s nulovým součtem. Při výpočtech, popsaných v kapitole 2.4, je při klasickém řešení pomocí tužky a papíru velmi snadné dopustit se chyby, která může mít značný dopad na výsledné řešení. Dále i při správném řešení, kdy jsou výsledkem vektory optimálních strategií hráčů a cena hry, není prakticky možné ověřit, zda jsou vektory strategií a hodnota ceny hry správné, jelikož je k jejich ověření zapotřebí značné množství opakování hry. Především na tyto nevýhody se nový systém zaměřuje a eliminuje je.

3.1.2 K ČEMU bude systém sloužit

IS bude umožňovat pochopit metody řešení používané při výpočtech ryzích a smíšených strategií pomocí příkladů, na nichž nejen přehledně zobrazí postup řešení pomocí různých metod, ale také umožní si vybrané hry zahrát, díky čemuž prakticky ověří správnost vypočtených řešení.

3.1.3 KDO bude se systémem pracovat

Systém je určen především pro studenty předmětu Teorie her, kterým umožní lepší pochopení předmětu. Dále také pedagogům tohoto předmětu pro účely kontroly správnosti výpočtů studentů při opravování zápočtových a zkouškových písemných prací. Jelikož není nutné se do systému přihlašovat a jedná se o internetovou aplikaci, může jej využívat také široká veřejnost (při nasazení aplikace na veřejně přístupný server).

3.1.4 VSTUPY a VÝSTUPY systému

Vstupy systému budou nastavení typu hry, výplatní matice, Kuhnova stromu, použití dominování a nastavení simulované hry. Výstupy pak podrobná řešení hry pomocí různých metod, popsaný postup odstranění dominovaných strategií, hledání sedlového bodu, výsledky simulované hry dvou počítačových hráčů a hry uživatele proti počítači.

3.2 Nefunkční požadavky

Jelikož je výsledným systémem internetová aplikace, je k nasazení nutný server s podporou technologie ASP.NET. Tato aplikace se musí korektně zobrazovat minimálně v 5 nejpoužívanějších internetových prohlížečích[3], což jsou v současné době Google Chrome, Mozilla Firefox, MS Internet Explorer, Apple Safari a Opera. Testování proběhne pouze v aktuálních nejvyšších verzích těchto prohlížečů.

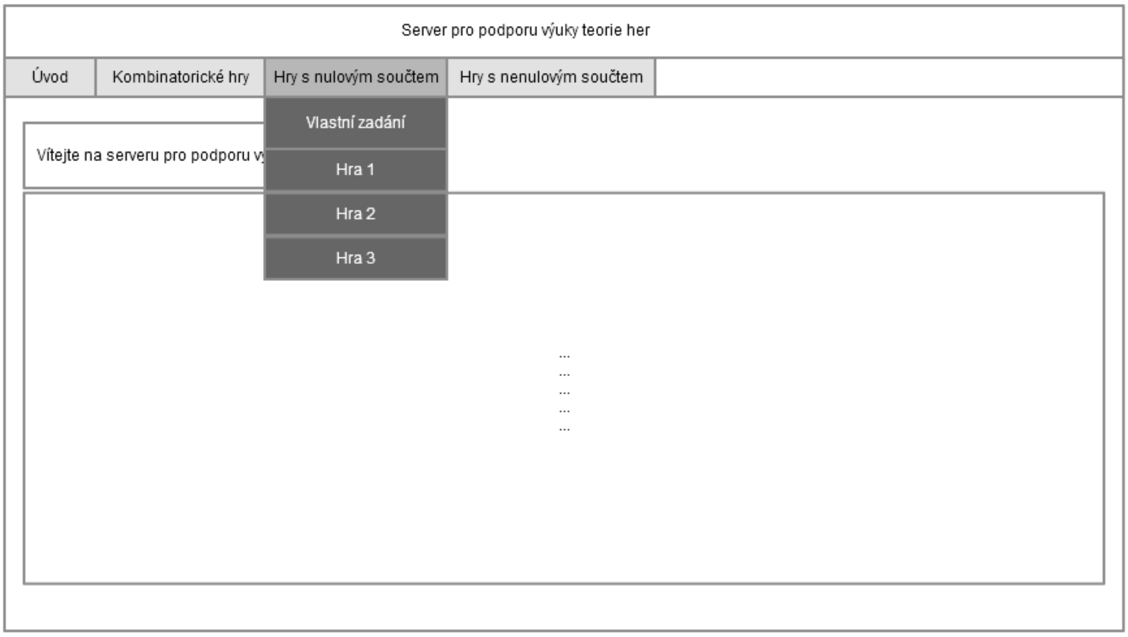
Hlavní důraz bude kladen na přehlednost stránek, rychlou odezvu a srozumitelnost řešení pomocí jednotlivých metod.

4 Návrh

Tato kapitola popisuje návrh aplikace jak z pohledu funkčnosti, definované v kapitole 3, tak hlediska jejího vzhledu.

4.1 Základní uživatelské rozhraní

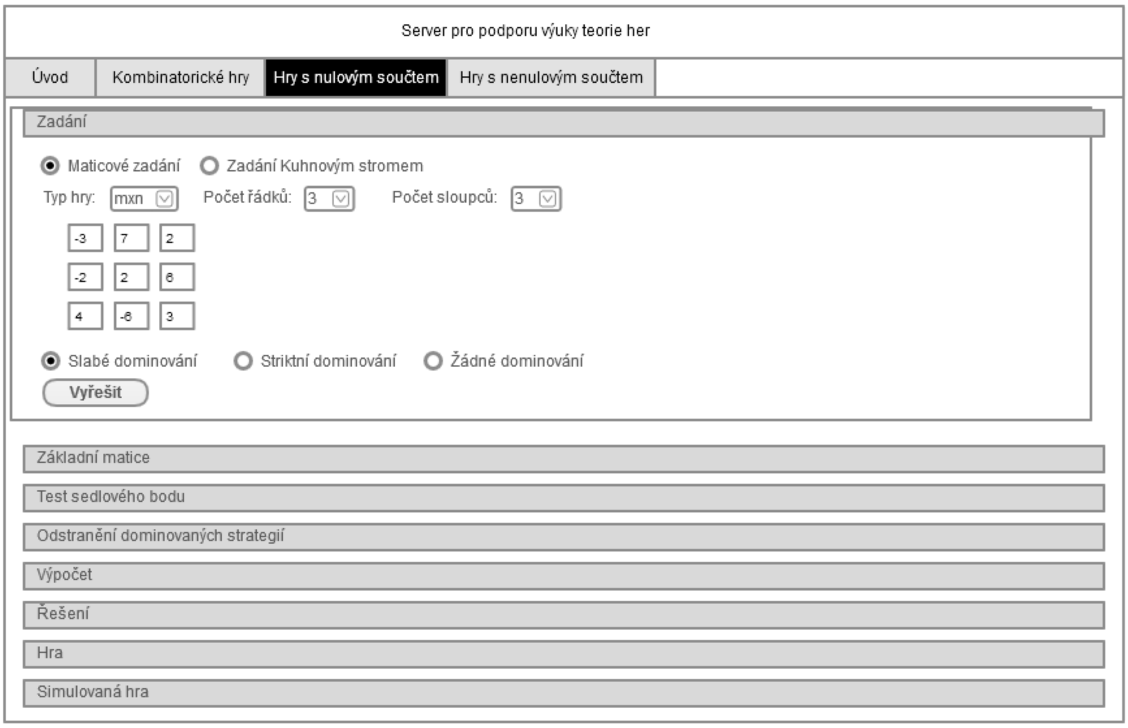
Po dohodě s kolegou Bc. Markem Záškodným, který zpracovával část her s nenulovým součtem, jsme navrhli základní vzhled aplikace, viz Obr. 15. Základní menu, umístěné v horní části aplikačního okna, rozděluje IS mezi 3 hlavní části teorie her probírané v předmětu „Teorie her a modely rozhodování v podmínkách neurčitosti". Kromě těchto 3 částí se zde ještě nachází odkaz na úvodní stranu, kde jsou popsány základní informace o tomto IS. Mezi zmíněné 3 části aplikace patří hry kombinatorické, hry s nulovým součtem a hry s nenulovým součtem. První část, tedy hry kombinatorické, byla již vytvořena v minulosti Ing. Tomášem Bařákem [4]. Jelikož je tato část implementována pomocí odlišné architektury, je zde uveden pouze hypertextový odkaz na současné umístění této aplikace. Druhá část, zaměřená na hry s nulovým součtem, byla již hlavním cílem této diplomové práce. Rozbalovací menu této části je opět k vidění na Obr. 15. První položka má název „Vlastní zadání". Umožňuje ručně zadat vstupní data hry. Následující položky již představují vytvořené hry, které si mohou studenti zahrát a budou popsány v kapitole 5.3.7.



Obr. 15 - Celkový návrh vzhledu aplikace

4.2 Vlastní zadání hry

Stránka s názvem „Vlastní zadání“ představuje stěžejní část aplikace. V horní části stránky si uživatel v prvním kroku vybere, zda chce zadat vstupní data hry pomocí matice výplatních funkcí, nebo pomocí Kuhnova stromu. V případě maticového zadání zvolí typ matice, tedy 2×2, 2×n, m×2 nebo m×n. Poté určí, jakým způsobem mají být z matice odstraněny dominované strategie. Stisknutí tlačítka „Vyřešit“ v případě maticového zadání vyvolá validaci vstupních textových polí, při které je kontrolováno, zda jsou zadané údaje v požadovaném formátu a rozsahu. Pokud je validace úspěšná, je stránka zpětně odeslána na server, kde proběhnou příslušné výpočty. Následně je zobrazena stránka s podrobným výpočtem členěným dle obsahu do panelů, jak je vidět na Obr. 16. Kromě panelu *Zadání* jsou to panely *Základní matice* obsahující matici vytvořenou pomocí zadaných hodnot, dále panel s názvem *Test sedlového bodu* s postupem hledání sedlového bodu v matici, panel *Odstranění dominovaných strategií*, v němž je zobrazen podrobný postup eliminace dominovaných strategií podle zvolených kritérií. Dále panel *Výpočet* zobrazující výpočty optimálních smíšených strategií a ceny hry za použití metod popsaných v kapitole 2.4, panel *Řešení*, v němž nalezneme výsledné vypočtené vektory ryzích a smíšených strategií obou hráčů a cenu hry, panel *Hra*, v němž je možné zahrát si zadanou hru s počítačovým soupeřem, a panel *Simulovaná hra* sloužící k simulaci zvoleného počtu opakování hry dvou počítačových hráčů.



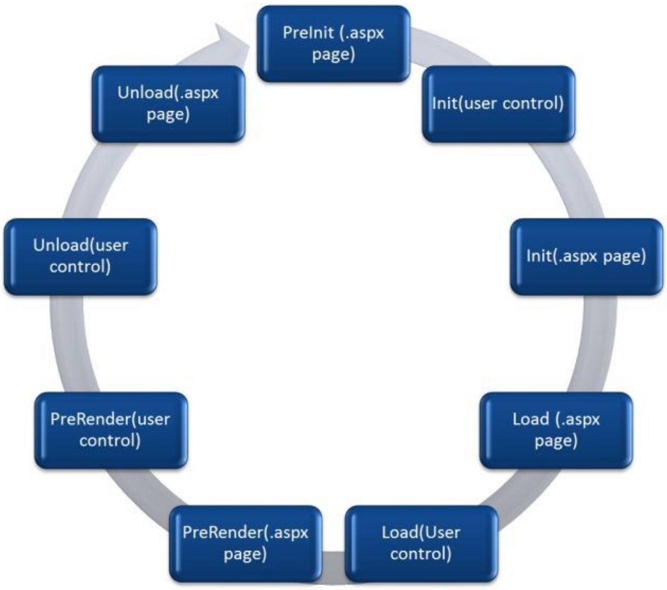
Obr. 16 - Návrh vzhledu aplikace pro hry s nulovým součtem

5 Implementace

Implementačním prostředím byla zvolena technologie ASP.NET realizována pomocí komponent WebForms, přičemž byl jako implementační jazyk použit C#.

5.1 Technologie ASP.NET

Technologie ASP.NET společnosti Microsoft je založena na CLR (Common Language Runtime), díky čemuž je možné psát zdrojové kódy aplikace v libovolném jazyce z frameworku .NET a rovněž využívat knihovny pocházející z tohoto frameworku. Programování WebForms (webových formulářů) se velmi podobá programování desktopových aplikací typu Windows Forms. To znamená, že stránky jsou poskládány z jednotlivých komponent o různých vlastnostech, které mohou vyvolávat různé události. Při požadavku prohlížeče na stránku ASP.NET překladač spustí a zkompiluje veškeré požadované skripty a následně prohlížeči vrátí stránku ve formátu HTML. Zpracování zaslaného požadavku a generování stránky prochází několika fázemi. Fáze, během nichž je možné změnit vzhled a nastavení prvků stránky, zobrazuje Obr. 17.



Obr. 17 - Technologie ASP.NET

PreInit – tato fáze nastává ještě předtím, než se začne stránka načítat. Je zde krom jiného možné ověřit, zda jde o první zobrazení stránky nebo se jedná o opětovné poslání (postback), dále lze dynamicky upravit prvky stránky či změnit vlastnosti MasterPage.

Init – poté, co jsou všechny prvky stránky inicializovány, včetně aplikace jejich vzhledu, je spuštěna tato fáze. V této fázi je možné číst a inicializovat vlastnosti jednotlivých prvků stránky.

Init je možné rozdělit do 2 skupin událostí – do inicializace celé strany a inicializace jednotlivých prvků stránky. Inicializace jednotlivých prvků je spuštěna před inicializací stránky.

Load – v této fázi stránka vyvolává svou metodu OnLoad a poté rekurzivně volá metodu OnLoad pro všechny své potomky až do doby, kdy jsou všichni tito potomci načteni. To je zachyceno v diagramu (Obr. 17), kdy je nejprve volána událost Load stránky a až poté jednotlivých prvků.

PreRender – nastává poté, co jsou vytvořeny všichni potomci stránky. Tato fáze je poslední, kdy je možné změnit nastavení prvků předtím, než budou vykresleny. Nejprve je opět volána tato metoda pro stránku a až poté pro její prvky.

UnLoad – tato fáze je určena k vyčištění vybraných prvků, jako například uzavření databázových připojení či uzavření otevřených souborů. Během této fáze jsou prvky stránky zobrazovány, proto je již není možné nijak měnit.

5.2 Ajax Control Toolkit

Ajax Control Toolkit [3] je rozšíření ASP.NET, přinášející nové nástroje umožňující tvorbu uživatelsky lépe vypadajících webových aplikací. Každý z nástrojů má kromě svých specifických vlastností k dispozici nastavení vzhledu pomocí třídy kaskádových stylů (css), díky němuž je možné nastavit vzhled prvku podle potřeb. Elementy tohoto balíku nástrojů využívají javascript, díky němuž pracují bez nutnosti opětovného poslání stránky na server (postback). V současné verzi z prosince 2013 (verze 4.1.7.1213) obsahuje více než 40 nástrojů. Jelikož však tato verze obsahovala v klíčových nástrojích chyby a nedostatky, byla pro implementaci IS zvolena stabilnější verze z června 2013 (verze 4.1.7.607). Konkrétně byly použity nástroje *TabContainer*, *CollapsiblePanel*, *DragPanel*, *ModalPopup*, *Slider* a *LineChart*.

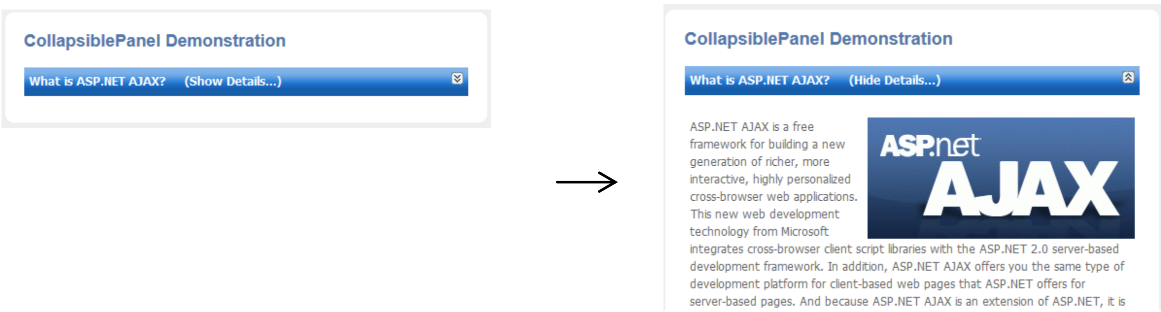
5.2.1 TabContainer

Nástroj *TabContainer* umožňuje organizovat obsah webové stránky do záložek podobně, jako je tomu v případě nástroje *TabControl* u desktopových aplikací využívajících Windows Forms. *TabContainer* sdružuje prvky typu *TabPanel* reprezentující jednotlivé záložky, jejichž obsah je definován párovým tagem *ContentTemplate*. Díky použití *ViewState* si aplikace při operaci *postback* pamatuje naposledy aktivní záložku, takže ji není nutné nastavovat ručně. Ukázku tohoto nástroje, použitého v aplikaci, je možné vidět na Obr. 18.

Obr. 18 - TabContainer

5.2.2 CollapsiblePanel

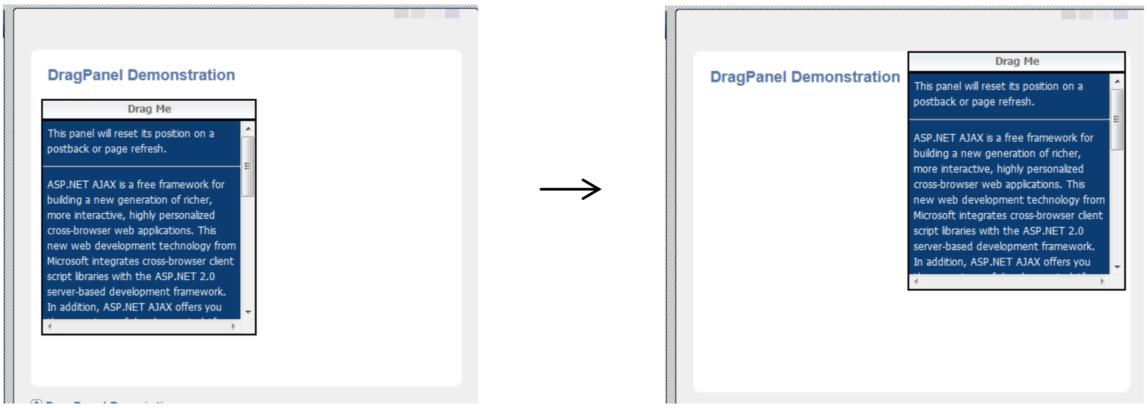
Tento nástroj umožňuje dynamicky skrývat a zobrazovat části stránek, což vede ke zvýšení přehlednosti. Jeho parametr *TargetControlID* označuje komponentu typu *asp:Panel*, jejíž viditelnost bude zobrazovat, případně skrývat. Zobrazení a skrytí panelu bývá nejčastěji realizováno jednou komponentou, při jejíž aktivaci (kliknutí) dojde k požadované transformaci. Tato komponenta je přiřazena do atributu *ExpandControlID*, případně *CollapseControlID*. Pomocí atributu *ImageControlID* a *TextLabelID* je rovněž možné zvolit komponenty, jejichž hodnota se bude měnit v závislosti na stavu panelu, kde *ImageControlID* je identifikátor objektu typu *asp:Image* a *TextLabelID* je identifikátor objektu typu *asp:Label*. Obr. 19 ukazuje nejprve skrytý panel pomocí *CollapsiblePanel* a poté rozbalený.



Obr. 19 - CollapsiblePanel

5.2.3 DragPanel

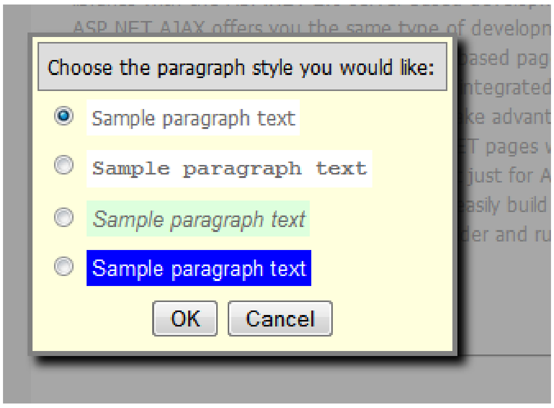
Nástroj *DragPanel* přibližuje chování okna na webové stránce oknu desktopové aplikace. To znamená, že je možné vybrané okno pomocí myši libovolně přemisťovat po stránce. To je přínosné, zejména pokud okno překrývá důležitou část obsahu stránky. Komponenta *DragPanel* mnoho nastavitelných vlastností nemá – pouze atribut *TargetControlID* pro nastavení identifikátoru objektu, jenž se má stát posouvatelným a atribut *DragHandleID*, po jehož uchopení (najetí kurzorem myši na jeho pozici, kliknutí a táhnutí) dojde k posunu objektu, specifikovaného atributem *TargetControlID*. Ukázka použití *DragPanelu* je na Obr. 20.



Obr. 20 - DragPanel

5.2.4 ModalPopup

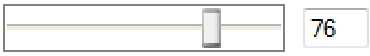
Nástroj s označením *ModalPopup* je určen k zobrazení nového okna, přičemž dojde k deaktivaci původního obsahu stránky. To je užitečné, pokud potřebujeme uživateli například sdělit zprávu, nebo se jej dotázat na další postup. Nejdůležitějšími atributy jsou: *PopupControlID*, jímž nastavujeme identifikátor objektu (panelu), který bude zobrazen po aktivaci, dále *TargetControlID* představující identifikátor objektu, jenž zobrazí panel nastavený atributem *PopupControlID*, poté *OkControlID*, což je identifikátor objektu, který způsobí zavření okna s pozitivním výsledkem a *CancelControlID*, jenž způsobí zavření okna s negativním výsledkem. Tento prvek je k vidění na Obr. 21.



Obr. 21 - ModalPopup

5.2.5 Slider

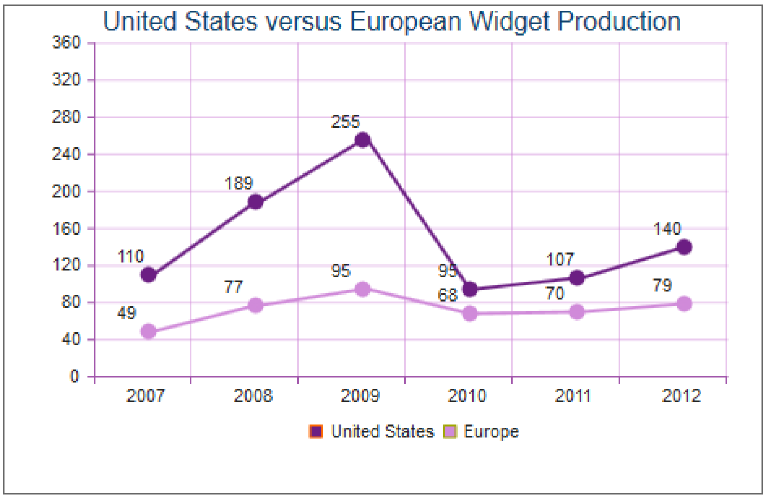
Slider je nástroj, pomocí něhož je možné velmi rychle a snadno vybrat číslo v požadovaném rozsahu. Atributy *Minimum* a *Maximum* se nastaví hranice rozsahu a atributem *Steps* počet hodnot mezi hranicemi *Minimum* a *Maximum*. Podobně jako většina nástrojů tohoto toolkitu obsahuje *Slider* atribut *TargetControlID*, jímž je nutné nastavit identifikátor objektu, který bude uchovávat nastavenou hodnotu této komponenty. Dále obsahuje atribut *BoundControlID* sloužící k zobrazení aktuální nastavené hodnoty. Tento nástroj je zobrazen na Obr. 22.



Obr. 22 - Slider

5.2.6 LineChart

Posledním použitým nástrojem z *Ajax Control Toolkitu* je *LineChart*. Jak už napovídá název, zobrazuje tento nástroj spojnicový graf. Pomocí párového tagu *Series* se určí jednotlivé sady bodů, z nichž budou vykresleny spojnice a také názvy a barevná provedení. Důležitým atributem, který musí být vyplněn, je *CategoriesAxis* typu *String*, jehož hodnoty popisují horizontální osu grafu. Dále je možné nastavit celkový vzhled grafu pomocí kaskádových stylů, název grafu a další méně důležité atributy. Ukázka spojnicového grafu je na Obr. 23.



Obr. 23 - LineChart

5.3 Aplikace

Jelikož vznikala část aplikace pro hry s nulovým součtem souběžně s částí pro hry s nenulovým součtem [1], bylo celé programové řešení (solution) rozděleno do 3 projektů: knihovny tříd pro hry s nulovým součtem, knihovny tříd pro hry s nenulovým součtem a knihovny uživatelského rozhraní typu webová aplikace, kterou používají obě knihovny tříd. Protože aplikace vznikala ve frameworku .NET za pomocí nástroje MS Visual Studio, nebyla tvorba uživatelského rozhraní obtížná. Byla vytvořena základní stránka (masterpage), z níž všechny ostatní stránky dědí použitý design.

Každý z panelů využívá nástroje *CollapsiblePanel* (kapitola 5.2.2), díky čemuž je možné jednotlivé panely zobrazovat a skrývat, což značně zpřehledňuje celkový vzhled stránky.

5.3.1 Zadání vstupních parametrů hry

Jediným panelem, který je při prvním zobrazení stránky viditelný, je panel s názvem „Zadání“. Tento panel umožňuje zadat a zobrazit výpočet hry, zadané pomocí vlastních hodnot. Ve své horní části obsahuje přepínač, který určuje, zda budou vstupní parametry hry zadány výplatní maticí, či Kuhnovým stromem. Při použití maticového zadání uživatel nejprve vybere typ matice (2×2, 2×n, m×2, m×n). Změna vybraného typu rozměru způsobí opětovné odeslání stránky na server (postback), díky němuž jsou podle typu výběru zobrazeny možnosti nastavení rozměrů matice (tedy např. u typu 2×n nastavení počtu sloupců, u typu m×2 počtu řádků atd.). Při změně tohoto nastavení je opět proveden postback a následně zobrazen požadovaný počet polí matice. Nyní může uživatel zadávat vstupní hodnoty výplatní matice, přičemž tyto hodnoty musí být celá čísla v intervalu ⟨-499; 499⟩. Interval hodnot byl zvolen poměrně nízký, jelikož je aplikace primárně určena pro výukové účely, kde je kladen důraz na vzhled, přehlednost a rychlost výpočtu. Při zadání neplatné hodnoty je uživateli zobrazena informace o nesprávném zadání a je mu zamezeno odeslání nesprávných hodnot zpět na server. Po zadání validních údajů je uživateli umožněno zvolit si, jakým způsobem má program při výpočtu pracovat s dominovanými strategiemi. Konkrétně má na výběr ze 3 možností: „Slabé dominování“ (viz kapitola 2.3.6.2), „Striktní dominováni“ (2.3.6.1) a možnost dominované strategie ponechat, tedy „Žádné dominování“.

Druhá možnost zadání hry je pomocí Kuhnova stromu. Tato metoda byla převzata od kolegy Bc. Marka Záškodného [1] a mírně upravena. Uživatel si z rozbalovacího menu vybere objekt, jenž chce do Kuhnova stromu přidat. Na výběr má z 3 možností – „Hráč I“, „Hráč II“ a „Náhoda“. Po výběru objektu z tohoto seznamu jej do stromu přidá pomocí tlačítka s popiskem „+“, které se nachází na pozici každého listu stromu. Stisknutí na toto tlačítko způsobí opětovné odeslání stránky na server, přičemž je vygenerována nová stránka s aktualizovaným počtem tlačítek a obrázkem pozadí propojující jednotlivá tlačítka do stromu. Jakmile je uživatel hotov s tvorbou stromu, zadá hodnoty jednotlivých listů stisknutím tlačítka „Ohodnotit strom“. Po jeho stisknutí dojde k nahrazení tlačítek textovými poli, jejichž hodnoty představují výplatní funkce pro jednotlivé listy stromu. Mimo toto nastavení je také důležité zvolit informační množiny těchto uzlů. To je realizováno rozbalovacím menu, které se nachází pod každým uzlem. V případě přidání objektu „Náhoda“ je při zadávání hodnot prostřednictvím textových polí uživateli umožněno zadat pravděpodobnosti výběru uzlů následujících za tímto objektem. Po zadání vstupních hodnot aplikace zkontroluje, zda jsou v případě objektů typu „Náhoda“ sumy pravděpodobností u jednotlivých uzlů vždy rovny 1 a také, zda mají uzly ve stejných informačních množinách stejný počet potomků.

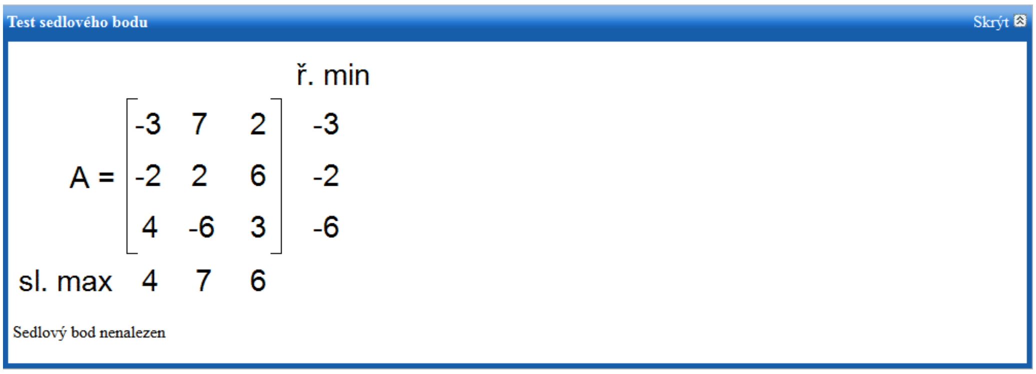
Následující kapitoly ukazují výstup aplikace pro příklad 4, jehož výplatní matice je na Obr. 6. Stisknutí tlačítka „Vyřešit“ způsobí u obou metod zadání zpětné odeslání na server, přičemž, pokud jsou zadaná data platná, dojde k výpočtu a zobrazení ostatních panelů popisujících tento výpočet.

5.3.2 Příprava řešení hry

Jelikož již jsou vstupní parametry hry zadány (kapitola 5.3.1), může aplikace přistoupit k vlastnímu výpočtu. Nejprve však v panelu nazvaném „Základní matice“ (Obr. 24) uživateli zobrazí výplatní matici, která vznikla ze vstupních hodnot zadaných pomocí matice nebo Kuhnova stromu. Tato matice je zobrazena v bitmapovém formátu (.gif)– z důvodu jejího snadnějšího zobrazení webovým prohlížečem.

Obr. 24 - Panel „Základní matice“

Dalším panelem je panel označený „Test sedlového bodu“. Jak již napovídá jeho název, úkolem tohoto panelu je rozhodnout, zda výplatní matice obsahuje nebo neobsahuje sedlový bod (viz kapitola 2.3.1). Tento test je realizován metodou `FindSaddlePoint()` třídy `Game` vracející objekt typu `SaddlePoint`, což je jednoduchý objekt, mající vlastnosti `X`, `Y` a `Value`, které představují pozici tohoto bodu v matici a jeho hodnotu. Po dokončení tohoto testu je výsledek opět zobrazen pomocí bitmapového obrázku, přičemž jsou v tomto obrázku kromě základní matice zapsány minimální hodnoty řádků a maximální hodnoty sloupců matice, díky čemuž je zřetelnější, jakým způsobem byla přítomnost sedlového bodu testována. Pokud matice žádný sedlový bod neobsahuje, zobrazí aplikace informaci o nenalezení sedlového bodu a výpočet pokračuje dále. Pokud však nalezen byl, je tento bod v matici zvýrazněn červenou barvou, dále je o této skutečnosti uživateli zobrazena zpráva a výpočet tímto končí. Kromě zprávy o nalezení je také zobrazena hodnota ceny hry, která odpovídá hodnotě sedlového bodu. Na Obr. 25 je ukázka panelu „Test sedlového bodu“ pro případ zadání, které sedlový bod neobsahuje.



Obr. 25 - Panel „Test sedlového bodu“

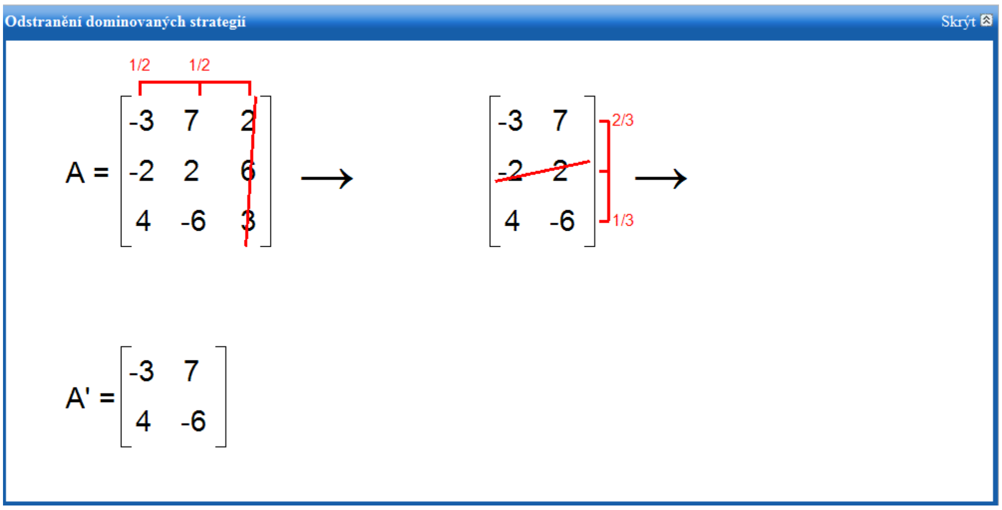
Pokud nebyl sedlový bod nalezen, je zobrazen panel popisující postup odstranění dominovaných strategií. Ten zobrazuje dominování podle nastavení, provedeném v panelu „Zadání" (kapitola 5.3.1). Pokud zvolil uživatel možnost „Žádné dominování", neobsahuje tento panel žádný postup. V případě možností „Slabé dominování" nebo „Striktní dominování" je postup zobrazen opět pomocí bitmapových obrázků. Odstranění dominovaných strategií má na starost metoda `Dominate(String path)` třídy `Game`. Parametr `path` zde zprostředkovává přístup ke složce náležící projektu uživatelského rozhraní, do níž vytvořené obrázky ukládá. Tato metoda pracuje v několika cyklech, v prvním nejprve hledá dominované sloupce. Pokud takovýto sloupec nalezne, vytvoří obrázek zachycující tuto skutečnost a odstraní tento sloupec posunutím hodnot matice doleva a zmenšením jejího rozměru o 1. Pro co nejzřetelnější vyobrazení byl zvolen způsob přešrktnutí dominované strategie červenou úsečkou, jak ukazuje Obr. 26. Jakmile již nejsou žádné další dominované sloupce nalezeny, pokračuje algoritmus hledáním dominovaných řádků podobným principem, jako postupoval u hledání dominovaných sloupců. Pokud při hledání dominovaných řádků nalezl alespoň jeden takovýto řádek, vrací se algoritmus na začátek, tedy k hledání dominovaných sloupců. To činí z důvodu, že odstraněním dominovaného řádku matice může obsahovat sloupce, které před odstraněním dominovaného řádku dominovány nebyly, ale nyní již dominovány jsou. Tento postup se opakuje tak dlouho, dokud je při každé iteraci nalezen alespoň jeden dominovaný řádek nebo dominovaný sloupec. Jakmile již matice žádné takovéto strategie neobsahuje, je přistoupeno k další metodě odstraňování dominovaných strategií. Jedná se o složené dominování, které bylo popsáno v kapitole 2.3.6.3. Ve zkratce je to situace, kdy je jedna strategie dominována kombinací několika jiných strategií. Z výkonnostních důvodů byla zvolena metoda testování kombinací 2 strategií, kdy jsou zkoušeny všechny kombinace získané z následujícího vzorce, kde první člen představuje násobný koeficient první strategie a druhý člen násobný koeficient druhé strategie:

$$\frac{x}{n}, \frac{n-x}{n}$$

kde $n \in \langle 2; n \rangle$, $x \in \langle 1; n\text{-}1 \rangle$.

Celkový počet kombinací lze vypočítat z $\sum_{i=1}^{n} x_i$. Jako hraniční hodnota *n* byla zvolena hodnota 10, při které je počet testovaných kombinací 45. Pokud některá z těchto 45 kombinací první a druhé strategie dominuje nad strategií třetí, je opět vytvořen obrázek demonstrující tuto dominanci, ve kterém jsou vyznačeny kromě strategií účastnících se tohoto dominování také jejich násobné koeficienty. Algoritmus nejprve nastaví koeficienty podle výše popsaného postupu a následně se pokouší nalézt dominované sloupce. Pokud žádný takový nenalezne, pokouší se nalézt dominované řádky. Jestliže opět neuspěje, změní násobné koeficienty a postup opakuje. Jakmile je však takto dominovaná strategie nalezena, vrátí se násobné koeficienty na svou počáteční hodnotu a celý proces se opakuje.

Výsledné obrázky, z nichž každý zobrazuje odstranění jedné dominované strategie, jsou poté do tohoto panelu umístěny za sebe ve stejném pořadí, v jakém dominování probíhalo. Nově vzniklá matice, vzniknuvší z poslední dominance, je označena *A'*. Všechny další výpočty jsou pak již realizovány s touto maticí, namísto původní matice *A*. Tento postup je vyobrazen na Obr. 26.



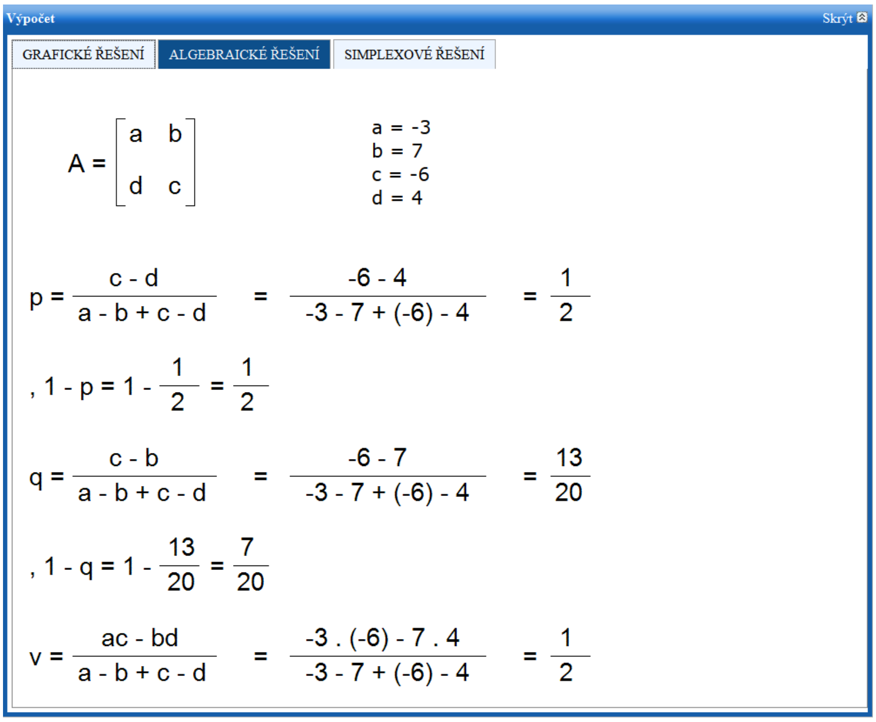
Obr. 26 - Panel „Odstranění dominovaných strategií“

5.3.3 Výpočet optimální smíšené strategie a ceny hry

Výpočet smíšených strategií hráče I a hráče II je realizován v panelu označeném „Výpočet“. Tento panel obsahuje nástroj typu *TabContainer* (kapitola 5.2.1), který aplikaci umožňuje přehledně zobrazovat všechny 3 metody výpočtu optimálních smíšených strategií. Jednotlivé záložky (nástroj *TabPanel*) jsou pojmenovány podle názvů těchto metod, tedy Algebraické řešení, Grafické řešení a Simplexové řešení.

5.3.3.1 Algebraické řešení

Jak bylo popsáno v kapitole 2.4.1 Hry typu 2×2, je algebraické řešení použitelné pouze u matic o rozměrech 2×2. Při splnění této podmínky je nejprve zobrazena obecná matice popisující polohy jednotlivých parametrů a, b, c, d a zobrazen popisek, který udává, jakých hodnot tyto parametry nabývají. Pro snadnější vykreslování výsledného algebraického řešení je vytvořeno 11 bitmapových obrázků (4 pro parametry p, q a 3 pro parametr v). První ze čtveřice obrázků pro parametr p a v zobrazuje obecnou rovnici výpočtu s parametry a, b, c, d, druhý pak dosazené hodnoty z matice A za tyto parametry, třetí výslednou hodnotu představující optimální pravděpodobnost volby první strategie hráčem a čtvrtý pravděpodobnost hraní druhé strategie hráčem. Pro zobrazení parametru v (ceny hry) postačují 3 obrázky, jelikož je výsledkem reálné číslo, nikoli vektor, jako u parametrů p a q. Důvodem rozdělení výpočtu do 4, respektive 3 samostatných obrázků, bylo správné vykreslování výpočtu v případě čísel o vyšším počtu míst. Tím, že jsou obrázky internetovým prohlížečem skládány za sebe, se při vyšší hodnotě šířky obrázku, nastávající u matic s vyššími hodnotami, mohou jednotlivé části výpočtu rozdělit na více řádků, díky čemuž je výpočet zobrazen korektně. Výsledný vzhled je zobrazen na Obr. 27.

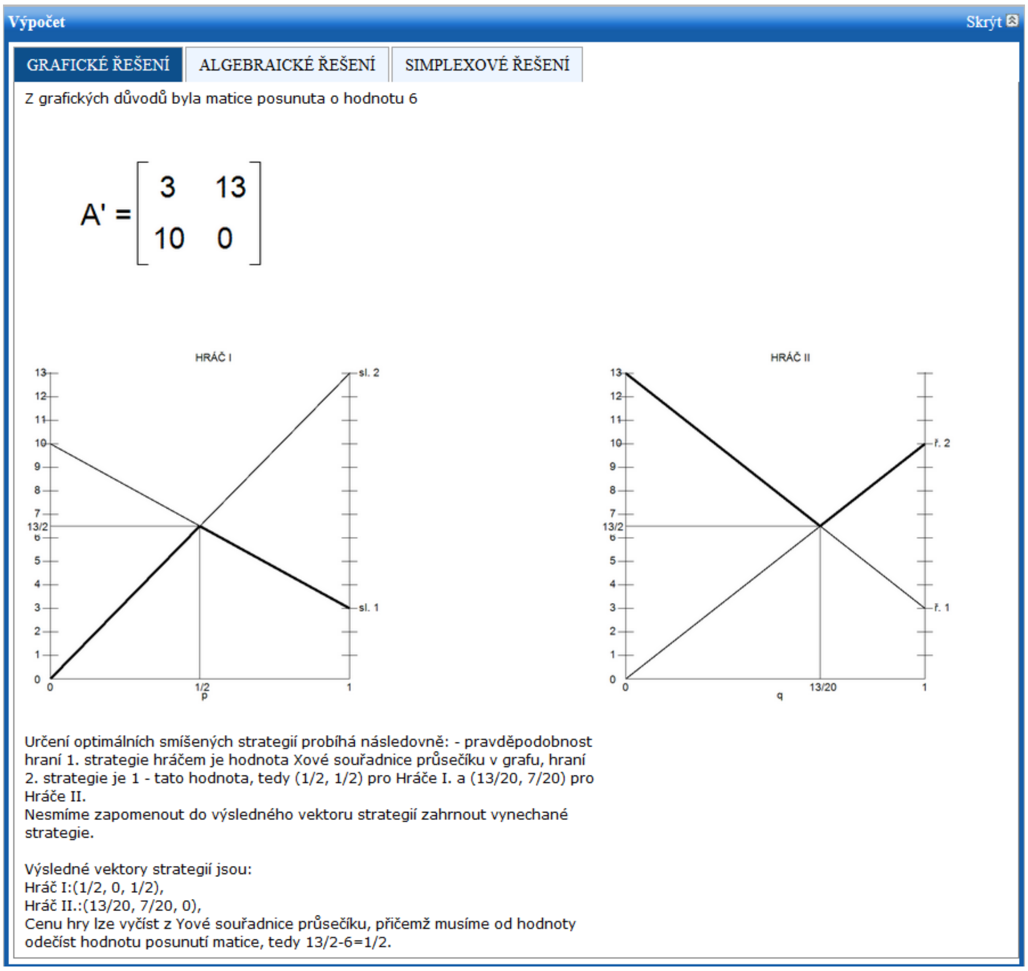


Obr. 27 - Algebraické řešení

5.3.3.2 Grafické řešení

Grafické řešení bylo popsáno v kapitole 2.4.2. Je vhodné pro matice výplatních funkcí, z nichž je alespoň jeden z rozměrů rovný 2. Výsledné řešení je realizováno pomocí bitmapového obrázku, z něhož je možné vyčíst optimální smíšené strategie hráčů. O vykreslení se stará metoda `DrawGraph()` třídy `Drawing`. Tato metoda přejímá mimo jiné parametr `values` představující dvourozměrné pole hodnot číselného typu double. Z důvodu čitelnějšího vykreslení jsou v případě, že matice obsahuje záporné hodnoty, všechny hodnoty matice navýšeny o konstantu c o velikosti nejnižší hodnoty matice A. Po základní inicializaci je vypočítáno měřítko vertikální osy tak, aby bylo výsledné grafické řešení co nejpřehlednější. Nyní program postupuje způsobem popsaným v kapitole 2.4.2. Nejprve jsou na levou vertikální osu vyneseny hodnoty ve vhodném měřítku z druhého řádku, respektive druhého sloupce matice výplatních funkcí. Následně jsou na pravou vertikální osu vyneseny hodnoty z prvního řádku, respektive druhého sloupce matice výplatních funkcí. Po spojení příslušných hodnot na těchto osách je nyní nutné nalézt dolní (hry 2×n), respektive horní (hry m×2) obálku grafu. To je realizováno průchodem průsečíků jednotlivých úseček zleva doprava, přičemž jsou hledány takové průsečíky, jejichž souřadnice y je nejnižší (v případě dolní obálky) nebo nejvyšší (v případě horní obálky). Následující postup se bude vztahovat k případu hledání dolní obálky. Nejprve jsou všechny úsečky, vyjádřené dvěma krajními body d_{nA} a d_{nB}, vyjádřeny obecnou rovnicí přímky (metoda `CountGeneralLineEquation()` třídy `Line`). Z těchto úseček je vybrána l_1, jež je určena 2 krajními body A_1 a B_1. Ta je z množiny všech úseček d_n vybrána tak, aby byla její hodnota A_{1Y} nejnižší. Poté je vyhledána úsečka l_2, která má s l_1 průsečík v požadovaném definičním oboru a současně je souřadnice X tohoto průsečíku ze všech průsečíků úsečky l_1 s úsečkami d_n nejnižší (tento průsečík je určen metodou `Intersect(Line`

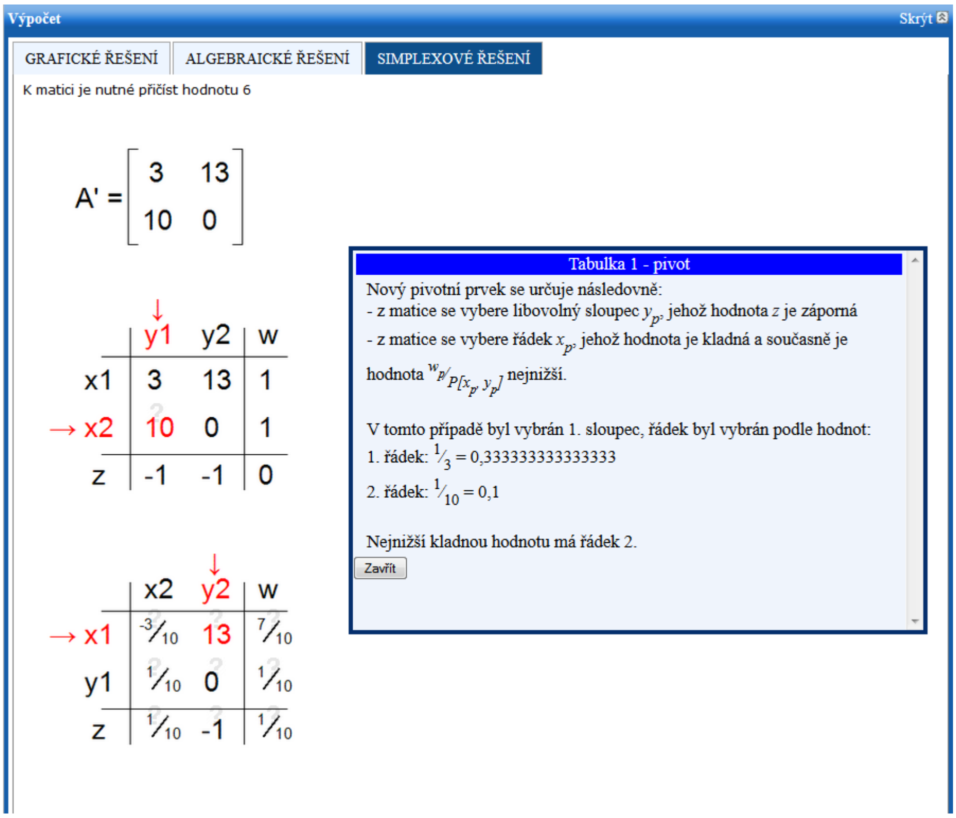
`line1, Line line2)` třídy `Drawing`). Následně je vytvořena nová úsečka l_3, kde počátečním bodem je průsečík nalezený v předchozím kroku, tedy $l_{3A} = l_1 \cap l_2$ a koncovým bodem je koncový bod úsečky l_2, tedy $l_{3B} = l_{2B}$. Nyní je opět hledána úsečka, tentokrát s označením l_4, jež má s úsečkou l_3 nejnižší hodnotu X ze všech úseček d_n. Po jejím nalezení je vytvořena nová úsečka l_5 mající počáteční bod v $l_3 \cap l_4$ a koncový bod v l_{4B}. Takto program pokračuje tak dlouho, dokud nachází nové průsečíky. Pokud je již dolní obálka grafu nalezena, je do grafu zanesena tučnou čarou, viz Obr. 11. Jelikož představuje dolní obálka minimální zaručený zisk hráče I, je nyní, pro dosažení co nejvyššího zisku, vyhledán nejvyšší bod této obálky (maximinový bod). Jeho souřadnice X je do grafu zapsána a představuje optimální pravděpodobnost hraní první strategie. Souřadnice Y tohoto bodu poté představuje cenu hry, jež je v grafu rovněž vyznačena. Pokud je matice typu 2×2, je vykreslen graf pro určení strategií druhého hráče způsobem podobným výše popsanému. V opačném případě je vytvořena nová matice označená A'' o rozměrech 2×2 obsahující strategie hráče II, které jsou získány z úseček vymezujících maximinový bod. Po vykreslení grafických řešení pro oba hráče je zobrazena informace, jak ze zobrazených grafů vyčíst smíšené strategie obou hráčů a cenu hry, viz Obr. 28.



Obr. 28 - Grafické řešení

5.3.3.3 Simplexové řešení

Řešení pomocí simplexové metody je jediným z implementovaných řešení, které je schopné řešit matice o libovolně zadaných rozměrech. S ohledem na postup popsaný v kapitole 2.4.3, je v případě, že matice obsahuje záporné hodnoty, přičtena ke všem hodnotám matice *A* konstanta *c* o velikosti nejnižší hodnoty matice. Poté je za pomoci třídy `Drawing` vykreslena první simplexová matice, v níž je červenou barvou zvýrazněn pivotní prvek. Po analýze, v jaké formě by bylo nejpřehlednější a nejpochopitelnější zobrazit způsob výpočtu hodnot simplexové tabulky, bylo vybráno zobrazení jednotlivých hodnot v samostatných panelech. Tyto panely jsou na stránku zobrazující výpočet přidávány dynamicky. K jejich zobrazení slouží tlačítka, jež se nacházejí nad jednotlivými hodnotami simplexové tabulky, viz Obr. 29. Tyto tlačítka mají nastavenu 90% průhlednost, díky čemuž nepřekážejí při čtení tabulky a jsou zvýrazněna pouze při přesunu ukazatele myši nad tato tlačítka. Aktivace vybraného tlačítka způsobí zobrazení panelu pomocí nástroje *ModalPopup*, popsaného v kapitole 5.2.4. Tento panel je navíc přesunovatelný, díky použití nástroje *DragPanel* (kapitola 5.2.3). Možnost panel přemisťovat byla zvolena z toho důvodu, že po zobrazení panelu s postupem výpočtu hodnoty prvku matice zakrývá tento panel původní matici, což k pochopení nepřispívá. Tím, že lze panel přesunout na libovolné místo na stránce, se však tento nedostatek odstraní. Nyní se vrátíme ke stavu, kdy je vykreslována první simplexová matice. Tato matice vznikne jednoduše dosazením hodnot ze zadané výplatní matice, což nepotřebuje nápovědy metodou zobrazení pomocných panelů. Co však již nemusí být zcela zřejmé, je určení pivotního prvku. Proto je na pozici prvku simplexové matice, určeného jako pivotní, zobrazena nápověda výše popsanou formou přesunovacího okna. Toto okno obsahuje podrobný postup, jakým byl pivotní prvek vybrán. Ukázka části tohoto panelu je vyobrazena na Obr. 29. Ostatní simplexové matice již obsahují nápovědy k výpočtu pro všechny své prvky. Po dokončení výpočtu, tedy zobrazení všech simplexových matic, je pomocí bitmapových obrázků zobrazeno výsledné řešení, což jsou vektory smíšených optimálních strategií a ceny hry. Jelikož je proces vytváření panelů, včetně jejich rozšíření, pro internetový prohlížeč docela výpočetně náročný, je zobrazení řešení pomocí simplexové metody omezeno na matice rozměru nejvýše 10×8, přičemž je při vyšším rozměru matice uživateli zobrazena informace o této skutečnosti.



Obr. 29 - Simplexové řešení

5.3.4 Řešení hry

Výsledné řešení hry, tedy vektory optimálních strategií obou hráčů a cena hry, je zobrazeno v panelu „Řešení“. Kromě optimálních smíšených strategií se zde také nachází řešení v oboru ryzích strategií. Postup hledání ryzích strategií není v žádném panelu popsán, jelikož je určení ryzích strategií snadno viditelné v panelu „Test sedlového bodu“ (kapitola 5.3.2). Pokud matice obsahuje více možných ryzích strategií, jsou zobrazeny všechny. Dále tento panel zobrazuje výše vypočtené vektory smíšených strategií ve stejné formě jako ryzí strategie a také cenu hry vyjádřenou jak zlomkem (u necelých čísel), tak číslem, zaokrouhleným na zvolený počet desetinných míst. Vzhled tohoto panelu je zobrazen na Obr. 30.

Obr. 30 - Panel „Řešení“

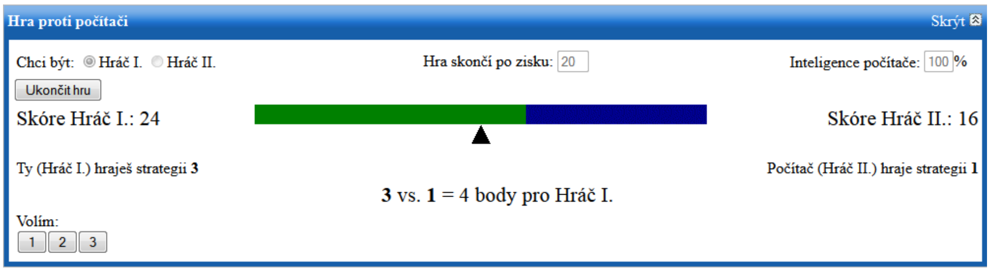
5.3.5 Hra proti počítači

Možnost zobrazit si výpočet a řešení zadané hry je pro uživatele sice přínosná, ale není nad to si zábavnou formou ověřit, zda vypočtené hodnoty odpovídají realitě. Z tohoto důvodu je zde panel s názvem „Hra proti počítači“, v němž je možné si nadefinovanou hru zahrát. Uživatel si nejprve zvolí, kterým z hráčů (hráč I nebo hráč II) chce být, tedy zda bude ve výplatní matici vybírat řádky či sloupce. Poté pomocí textového pole určí hodnotu, při jejímž překonání hra končí. Toto pole tedy umožňuje nastavit délku hry, jelikož čím je hodnota vyšší, tím vyšší počet kol je potřeba odehrát. Posledním nastavením před započetím hry je nastavení inteligence počítačového protihráče realizované prostřednictvím posuvníku (viz kapitola 5.2.5). Tato hodnota je vyjádřena v procentech, kde 100% znamená, že počítač hraje vždy podle vektoru svých optimálních strategií, a hodnota 0%, že své tahy volí náhodně. Uživatel si samozřejmě může zvolit jakékoli celé číslo v rozsahu 0 až 100. Stisknutí tlačítka „Spustit hru“ vytvoří novou hru ze zadaných parametrů, přičemž dojde k zobrazení aktuálního skóre obou hráčů, grafického ukazatele stavu hry a panelu možných tahů uživatele. Skóre obou hráčů je nastaveno na hodnotu, jež byla zadána v textovém poli pro nastavení bodů, při jejichž získání hra končí. Taktéž je grafický ukazatel stavu hry nastaven do rovnovážného stavu. Dále je vygenerováno tolik tlačítek, kolik má nepočítačový hráč strategií. Každé z těchto tlačítek, pojmenovaných podle pořadí strategií, symbolizuje jednu z těchto strategií. Kliknutím na vybranou strategii způsobí zpětné odeslání na server, kde je za pomoci metody `GetTurnStrategy` třídy `Game` vybrána strategie počítačového hráče v závislosti na jeho nastavené inteligenci. Hodnotu inteligence označíme a. Pokud je a vyšší než 0, je určeno pseudonáhodné reálné číslo b z intervalu $\langle 0; 1 \rangle$. Číslo a je převedeno do stejného formátu, jakého je b, tedy vydělením hodnotou 100. Pokud je b nižší než a, volí počítač svou optimální strategii, v opačném případě volí strategii náhodnou.

Pro $b \leq a$ postupuje metoda následovně: Nejprve je vygenerováno pseudonáhodné číslo c opět z intervalu $\langle 0; 1 \rangle$. Jelikož je součet hodnot z vektoru strategií roven 1, prochází program tyto strategie s od první do poslední, přičemž sčítá jejich hodnoty do proměnné d. Po každém navýšení d o hodnotu strategie s_i je otestováno, zda platí $c < d$. Pokud nerovnost platí, je vrácena poslední přičtená strategie, tedy s_i.

Pro $b > a$ je řešení snazší. Na počátku je vygenerováno pseudonáhodné číslo e z intervalu $\langle 0; n \rangle$, kde n je celkový počet strategií počítačového hráče. Následně je vrácena strategie, která má ve vektoru strategií (základním vektoru všech strategií počítačového hráče) pořadí n.

Po vrácení strategie počítačového hráče je provedeno vyhodnocení ceny hry pomocí zvolené strategie uživatele a strategie počítače. Obě tyto strategie jsou v panelu zobrazeny ve formátu *Zvolená strategie hráče I* vs. *Zvolená strategie hráče II = počet bodů* pro vítězného hráče. Také jsou zobrazeny strategie, jež oba hráči hrají, a také je aktualizován grafický ukazatel stavu hry, ve kterém zelená oblast představuje poměr bodů uživatele a modrá poměr počítačového hráče. Postupným volením strategií hra pokračuje až do doby, kdy je skóre jednoho z hráčů menší nebo rovno 0. Když tato situace nastane, je pomocí nástroje *ModalPopup* (kapitola 5.2.4) zobrazena informace o konci hry a vítězném hráči. Současně jsou také deaktivována tlačítka voleb strategie. Tlačítkem „Spustit hru“ je poté možné začít novou hru, přičemž dojde k aktivaci prvků, zajišťujících nastavení hry. Celý tento panel je zobrazen na Obr. 31.

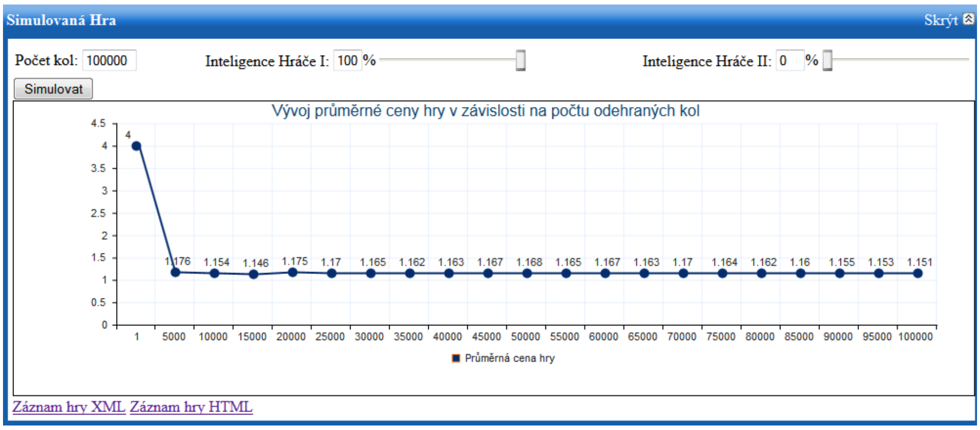


Obr. 31 - Panel „Hra proti počítači“

5.3.6 Simulovaná hra

Hraní hry proti počítači, uvedené v předchozí kapitole, umožňuje uživateli vyzkoušet si, zda hraní vypočtených optimálních strategií přinese vypočtenou průměrnou cenu hry. Jelikož se ale skutečná průměrná cena hry může přiblížit vypočtené až po značném množství kol, byl přidán panel s názvem „Simulovaná hra“, který uživateli tuto možnost nabízí, aniž by musel toto značné množství kol odehrát proti počítačovému protihráči. V horní části tohoto panelu se nachází textové pole, jehož hodnota udává, kolik kol proběhne mezi 2 počítačovými hráči. U obou těchto hráčů je možné nastavit inteligenci v % podobně, jako tomu bylo v případě hry uživatele proti počítači, tedy pomocí posuvníku (kapitola 5.2.5). Po stisknutí tlačítka s popiskem „Simulovat“ dojde ke spuštění animace vykreslující spojnicový graf ze zadaných hodnot. Horizontální osa reprezentuje jednotlivá kola hry, vertikální osa pak průměrnou cenu hry. Jelikož by při velkém počtu hracích kol bylo značně nepřehledné zobrazit průměrnou cenu hry pro každé kolo, byl zvoleno zobrazení pouze každé x.té hodnoty tak, aby horizontální osa obsahovala nejvýše 21 hodnot, při kterých je graf dobře čitelný. Ukázka simulované hry pro 100 000 kol a různá nastavení inteligence jsou k vidění na Obr. 32 a Obr. 33.

Obr. 32 - Panel „Simulovaná hra“ s inteligencí 100 %



Obr. 33 - Panel „Simulovaná hra“ s inteligencí 0 %

Po grafickém zobrazení simulované hry je možné si průběh hry zobrazit ve formátu XML či HTML, přičemž je z prostorových důvodů tato možnost omezena na maximálně 100 000 herních kol. Znamená to tedy, že pokud je zadaný počet herních kol vyšší než 100 000, nejsou volby zobrazení průběhu hry viditelné. Průběh simulované hry obsahuje pro každé kolo identifikátor tohoto kola (pořadí kola), strategii hráče I a hráče II, cenu hry v tomto kole a průměrnou cenu hry od prvního kola do tohoto kola.

5.3.7 Hry

Mimo vlastního zadání pomocí matice výplatních funkcí nebo Kuhnovým stromem obsahuje aplikace 3 hry, které si může uživatel zahrát. Všechny tyto hry byly originálně vymyšleny pro potřeby tohoto podpůrného serveru. Tyto hry jsou umístěny na samostatných stránkách, přičemž obsahují při svém prvním zobrazení pouze jeden panel s názvem „Pravidla“. Zde jsou popsána pravidla hry, což je především návod, jakým způsobem vytvořit matici výplatních funkcí. Stejně jako u panelu „Hra“ na webové stránce s vlastním nastavením hry, je i zde možnost nastavit, za kterého hráče chce uživatel hrát, dále hodnotu, po jejímž obdržení hra končí a také možnost nastavit inteligenci počítačového protihráče. Po stisknutí tlačítka „Spustit hru“ dojde k zahájení hry a zobrazení dalších panelů. Konkrétně je to panel „Základní matice“ obsahující matici výplatních funkcí, vytvořenou podle pravidel, popsaných v panelu „Pravidla“. Následně je to panel „Řešení“, který obsahuje vypočtené vektory optimálních smíšených strategií obou hráčů a panel se samotnou hrou, podobný panelu „Hra proti počítači“ z kapitoly 5.3.5. Panel „Řešení“ je při zobrazení stránky sbalený, aby uživatele neodrazoval od vlastního výpočtu optimálních strategií. Po dosažení nastaveného počtu bodů jedním z hráčů dojde opět k zobrazení hlášení o vyhrávajícím hráči.

5.3.7.1 Souboj prvočísel

Hra s názvem „Souboj prvočísel“ je jednoduchá hra založená na volení čísel 1, 2 a 3. Oba hráči zvolí současně jedno z těchto čísel. Pokud je součet zvolených čísel prvočíslem, získává body hráč I, v opačném případě získává body hráč II. Body mají velikost součtu těchto dvou čísel. Optimální smíšené vektory strategií jsou pro oba hráče shodné: $\left({}^{1}/_{24}, {}^{13}/_{24}, {}^{5}/_{12}\right)$. Cena hry, pro zapsání matice výplatních funkcí z pohledu hráče I, je $1/_{24}$ z čehož vyplývá, že je tato hra výhodnější pro hráče I, pokud hraje výše uvedenou smíšenou strategii.

5.3.7.2 Dobývání hradu

Tato hra je inspirována fantasy motivy. Obsahuje 2 hráče – Obránce a Útočníka. Každý z hráčů má k dispozici 5 různých typů postav, přičemž společnými typy jsou *Pěšák* a *Lučištník*. Kromě těchto 2 typů má Obránce na výběr také postavy *Rytíř*, *Pes* a *Princezna*. Útočník má naopak k dispozici postavy *Kopiník*, *Katapult* a *Čaroděj*. Každá z těchto postav je vůči některým postavám protihráče silná (porazí je) a vůči jiným naopak slabá (je jimi poražena). Informace o vzájemných interakcích všech těchto postav jsou obsaženy v panelu „Pravidla“. Tyto interakce byly voleny tak, aby žádná postava nebyla dominována jinou postavou. Vektor optimální smíšené strategie pro hráče Obránce je $\left({}^{69}/_{142}, {}^{3}/_{130}, {}^{7}/_{20}, {}^{4}/_{195}, {}^{10}/_{83}\right)$, pro hráče Útočník je to následující vektor $\left({}^{37}/_{130}, {}^{61}/_{195}, {}^{3}/_{20}, {}^{3}/_{41}, {}^{7}/_{39}\right)$. Pokud matici výplatních funkcí zapíšeme z pohledu hráče Obránce, je cena hry $-{}^{13}/_{69}$, takže je hra výhodnější pro hráče Útočník. Tato výplatní matice je vyobrazena na Obr. 34, přičemž jsou zelenou barvou zvýrazněny hodnoty výhodné pro hráče Obránce.

$$
\begin{array}{c|c|c}
 & \begin{array}{ccccc} \text{Pěšák} & \text{Lučištník} & \text{Kopiník} & \text{Katapult} & \text{Čaroděj} \end{array} & \\
\begin{array}{c} \text{Pěšák} \\ \text{Lučištník} \\ \text{Rytíř} \\ \text{Pes} \\ \text{Princezna} \end{array} & \begin{bmatrix} 0 & -1 & 3 & -2 & -1 \\ 1 & 0 & -2 & 5 & -3 \\ 1 & 2 & -4 & 3 & -4 \\ 2 & -2 & 2 & -1 & -2 \\ -5 & -3 & -2 & -3 & 15 \end{bmatrix} & \text{O B R Á N C E} \\
 & \text{Ú T O Č N Í K} &
\end{array}
$$

Obr. 34 - Výplatní matice pro hru Dobývání hradu

5.3.7.3 Kerbindak

Kerbindak je stará karetní hra původem z Holandska určená pro 2 až 4 hráče. Hraje se s klasickými francouzskými kartami s hodnotami 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K, A, přičemž jsou karty J, Q, K, A převedeny na hodnoty 11, 12, 13, 14. Z pravidel pro tuto hru byla vytvořena pravidla upravená tak, aby byla použitelná pro hru dvou hráčů s nulovým součtem. Hra tedy obsahuje dva hráče – Srdce a Pik. Oba hráči volí současně jednu kartu své barvy. Pokud má karta hráče vybírajícího řádky (Srdce) hodnotu o 1 nižší, stejnou, o 1 vyšší nebo je celým násobkem nebo dělitelem karty hráče vybírajícího sloupce (Pik), získává body o velikosti součtu hodnot těchto dvou karet hráč Srdce, v opačném případě získává tyto body hráč Pik. Hra končí, pokud skóre jednoho z hráčů dosáhne hodnoty 0 nebo nižší. Pokud si necháme zobrazit vektory optimální smíšené strategie hráčů, zjistíme následující:

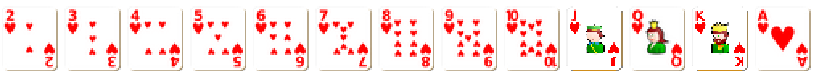
pro hráče Srdce je to vektor $\left(1/4, 0, 9/89, 0,0,0, 10/83, 0, 25/116, 0, 16/109, 1/140, 17/107\right)$, pro hráče Pik je to poté $\left(0,0,0, 3/13, 2/29, 1/123, 8/47, 26/135, 0,0,0, 49/178, 2/37\right)$.

Průměrná cena hry je $-679/141$, z čehož vyplývá, že je v této hře hráč Srdce oproti hráči Pik značně znevýhodněn. Výplatní matici pro tuto hru z pohledu hráče Srdce zobrazuje Obr. 35, na němž jsou zeleně zvýrazněny prvky výhodné pro tohoto hráče.

	2	3	4	5	6	7	8	9	10	J	Q	K	A
2	4	5	6	-7	8	-9	10	-11	12	-13	14	-15	16
3	5	6	7	-8	9	-10	-11	12	-13	-14	15	-16	-17
4	6	7	8	9	-10	-11	12	-13	-14	-15	16	-17	-18
5	-7	-8	9	10	11	-12	-13	-14	15	-16	-17	-18	-19
6	8	9	-10	11	12	13	-14	-15	-16	-17	18	-19	-20
7	-9	-10	-11	-12	13	14	15	-16	-17	-18	-19	-20	21
8	10	-11	12	-13	-14	15	16	17	-18	-19	-20	-21	-22
9	-11	12	-13	-14	-15	-16	17	18	19	-20	-21	-22	-23
10	12	-13	-14	15	-16	-17	-18	19	20	21	-22	-23	-24
J	-13	-14	-15	-16	-17	-18	-19	-20	21	22	23	-24	-25
Q	14	15	16	-17	18	-19	-20	-21	-22	23	24	25	-26
K	-15	-16	-17	-18	-19	-20	-21	-22	-23	-24	25	26	27
A	16	-17	-18	-19	-20	21	-22	-23	-24	-25	-26	27	28

Obr. 35 - Výplatní matice pro hru Kerbindak

Z důvodu větší přehlednosti byla zvolena grafická podoba jednotlivých strategií pomocí ikon jednotlivých karet. Hráč tedy v každém kole vybírá strategii stisknutím vybraného tlačítka, jež má podobu karty v hráčově barvě. Ukázka možností volby strategie hráče *Srdce* je k vidění na Obr. 36.



Obr. 36 - Ukázka vzhledu karet hry Kerbindak

5.3.8 Teorie

Stránka s názvem „Teorie" je pomocí nástroje *TabContainer* (kapitola 5.2.1) rozdělena do 2 částí. První část obsahuje veškerý teoretický základ, který je nutný k řešení her s nulovým součtem. Jsou zde popsány všechny pojmy z kapitol 2.1 a 2.3. Druhá část poté obsahuje ukázkové podrobné řešení maticově zadané hry typu 3×3 pomocí všech 3 metod popsaných v kapitole 2.4.

5.3.9 Uživatelská příručka

Uživatelská příručka je stránka, která obsahuje okomentované prvky stránky pro řešení her s nulovým součtem. Pomocí nástroje *TabContainer* (kapitola 5.2.1) je rozdělena do 2 částí. V první části jsou popsány elementy a vzhled stránky pro řešení vlastně zadané hry, ve druhé pak popsány předvytvořené hry. Z důvodu co nejvýstižnějšího popisu byla zvolena forma screenshotů jednotlivých částí stránek s číslovaným označením klíčových prvků zachycených na těchto snímcích. Tyto snímky jsou poté zobrazeny v jednoduché stránce, přičemž jsou prokládány texty o jejich funkčnostech a omezeních.

6 Nasazení

Pro správný chod aplikace je nutné ji umístit na server podporující technologii ASP.NET. Instalace této aplikace se provádí zkopírováním všech souborů a složek ze složky „Aplikace“ do příslušného adresáře webového serveru. Všechna nastavení jsou již obsažena v těchto souborech, proto není nutné žádná další nastavení provádět. Její spuštění je také možné na lokálním virtuálním serveru, jenž je součástí MS Visual Studia. K testovacím účelům byla aplikace umístěna na server českého poskytovatele webhostingu, společnosti ASPone, s.r.o. Výhodou tohoto poskytovatele je, že v základní verzi (program Webhosting FreeHosting) poskytuje tuto službu zdarma a bez jakýchkoli reklam. Jediným omezením je menší disková kapacita, než v případě placených verzí, a to 40 MB pro stránky a 30 MB pro databázi. Toto omezení však vyvinuté aplikaci postačuje.

Aplikace je umístěna na adrese http://teh-vsb.aspone.cz/, kde byla také využívána v průběhu testovací fáze. ASPone sice v současné době poskytuje tento hosting bezplatně, avšak není možné odhadnout, zda tomu tak bude i nadále.

Po většinu doby testování se aplikace chovala stejným způsobem, jako na lokálním virtuálním serveru. V posledních dnech se však objevila odlišnost v kvalitě dynamicky vytvářených bitmapových obrázků. Bitmapy, jež jsou vytvářeny na tomto serveru, mají přibližně o pětinu menší velikost než bitmapy vytvořené na lokálním serveru. Ztráta objemu dat je způsobena absencí vyhlazování, která je zejména u textu v těchto obrázcích (např. u matic výplatních funkcí) dosti viditelná. Třída `System.Drawing.Graphics`, kterou aplikace používá pro vykreslování obrázků, sice obsahuje několik nastavení kvality vykreslování, vyhlazování a antialiasingu, avšak žádné nastavení tento problém neodstranilo. Taktéž komunikace s podporou ASPone neprokázala, že na serveru došlo k uživatelsky neovlivnitelnému nastavení kvality výstupných obrázků.

7 Závěr

Cílem této diplomové práce bylo navrhnout a implementovat server, který bude sloužit studentům pro snadnější pochopení látky probírané v předmětu „Teorie her a modely rozhodování v podmínkách neurčitosti“. Toho mělo být docíleno pomocí jednotlivých komponent aplikace, kdy si studenti měli být schopni nejen zadat parametry, ale také zobrazit průběh řešení příkladu. Vzhledem k obsáhlému serverovému řešení, uživatelské příručce i postupům, které jsou popsané v této práci, můžeme říci, že byl cíl práce splněn. Jelikož jsem absolventem tohoto předmětu, mohl jsem vycházet z vlastních zkušeností s probíranou látkou a zaměřit se na témata, která při klasickém řešení na papír způsobovala mně i spolužákům největší problémy.

S kolegou Bc. Markem Záškodným jsme se dohodli na architektuře tohoto serveru, přičemž jsme navrhli společný design stránek tak, aby celá internetová aplikace působila jednotně. Mým úkolem bylo řešení maticově zadaných her, které jsem poskytl Markovi Záškodnému pro řešení her s nenulovým součtem, zatímco od něj jsem převzal funkcionalitu zadání hry pomocí Kuhnova stromu. Výsledná aplikace obsahuje 3 originální předvytvořené hry pro 2 hráče, které byly navrženy pro účely této diplomové práce. Uživateli je umožněno hrát za libovolného z těchto hráčů, přičemž může nastavovat inteligenci počítačového protihráče. Dále je možné vytvořit si hru pomocí vlastního zadání, kterou aplikace vyřeší a popíše podrobný postup řešení. Tuto uživatelsky vytvořenou hru je opět možné zahrát si s nastavením inteligence protihráče nebo nasimulovat průběh této hry o zadaném počtu kol 2 počítačových hráčů, u nichž je opět možné nastavovat inteligenci a následně si výsledný průběh zobrazit pomocí grafu ukazujícím vývoj průměrné ceny hry či zobrazit si tahy hráčů v jednotlivých kolech, taktéž s informací o průměrné ceně hry.

Věřím, že bude tato aplikace po svém nasazení na server využívána studenty předmětu „Teorie her a modely rozhodování v podmínkách neurčitosti“ a pomůže jim nejen s objasněním mezer v teorii, ale také jim umožní ověřit si díky implementovaným hrám správnost řešení v praxi.

Seznam obrázků:

Použité zdroje

[1] **ZÁŠKODNÝ, Marek.** *Diplomová práce - Server pro podporu výuky teorie her.* Ostrava : Vysoká škola báňská, 2014.

[2] **FERGUSON, Thomas S.** *Game Theory.* [Online] 2005. [Citace: 26. 3 2014.] http://www.math.ucla.edu/~tom/Game_Theory/mat.pdf.

[3] ASP.NET AJAX Control Toolkit. *ASP.NET AJAX Control Toolkit.* [Online] [Citace: 20. 4 2014.] http://www.ajaxcontroltoolkit.com/.

[4] www.w3schools.com. *Browser Statistics and Trends.* [Online] [Citace: 8. 4 2014.] http://www.w3schools.com/browsers/browsers_stats.asp.

[5] **BAŘÁK, Tomáš.** Server pro podporu výuky teorie her. *Diplomová práce - Server pro podporu výuky teorie her.* [Online] 2013. [Citace: 11. 4 2014.] http://dspace.vsb.cz/handle/10084/98534.

Seznam příloh

Příloha A – příloha na CD

- /Server_pro_podporu_vyuky_teorie_her/ - zdrojové soubory IS
- /Vysledna_aplikace/ - kompilovaná aplikace pro nasazení na server
- hur102.pdf - text diplomové práce
- dokumentace.xml – programátorská dokumentace